Patient-centered yes/no prognosis using learning machines

I.R. König

Institut für Medizinische Biometrie und Statistik,
Universität zu Lübeck,
Ratzeburger Allee 160, 23538 Lübeck, Germany

J.D. Malley and S. Pajevic

Center for Information Technology,
National Institutes of Health,
Bethesda, MD, USA

C. Weimar and H-C. Diener

Klinik und Poliklinik für Neurologie,
Universität Duisburg-Essen, Germany

A. Ziegler*

Institut für Medizinische Biometrie und Statistik,
Universität zu Lübeck,
Ratzeburger Allee 160, 23538 Lübeck, Germany
E-mail: ziegler@imbs.uni-luebeck.de
*Corresponding author

Abstract: In the last 15 years several machine learning approaches have been developed for classification and regression. In an intuitive manner we introduce the main ideas of classification and regression trees, support vector machines, bagging, boosting and random forests. We discuss differences in the use of machine learning in the biomedical community and the computer sciences. We propose methods for comparing machines on a sound statistical basis. Data from the German Stroke Study Collaboration is used for illustration. We compare the results from learning machines to those obtained by a published logistic regression and discuss similarities and differences.

Keywords: bagging; boosting; random forest; stroke; support vector machine; learning machines; data mining; bioinformatics.

Reference to this paper should be made as follows: König, I.R., Malley, J.D., Pajevic, S., Weimar, C., Diener, H-C. and Ziegler, A. (2008) 'Patient-centered yes/no prognosis using learning machines', *Int. J. Data Mining and Bioinformatics*, Vol. 2, No. 4, pp.289–341.

Biographical notes: Inke R. König received her PhD in Human Biology from the University at Lübeck, Germany. She received a Diploma in Psychology from the Philipps-University of Marburg, Germany. She is now a Scientific Employee at the Institute of Medical Biometry and Statistics, Medical Faculty, University at Lübeck, Germany. Her research interests include genetic epidemiology and clinical epidemiology. She has published over 90 refereed papers and is author of a standard text book on statistical approaches to genetic epidemiology published in English at Wiley-VCH.

James D. Malley, PhD has been with the National Institutes of Health since 1977, as a Research Mathematical Statistician. He does collaborative research with NIH scientists, and regularly teaches courses on data analysis. He has previously published two monographs (Springer-Verlag) on algebraic and statistical aspects of variance component analysis. His long-term interests include statistical genetics, and quantum computing. More recently he has conducted subject-matter collaborative research that applies statistical learning machines, and is writing a practical guide textbook on this subject (for Cambridge University Press, with coauthors K. Malley and S. Pajevic).

Sinisa Pajevic received his BS Degree in Electrical Engineering from the Belgrade University, Belgrade, Serbia, and his PhD Degree in Physics from the Boston University, Boston, Massachusetts. Since 1993, he has been working at the National Institutes of Health, Bethesda, MD, where he is currently a Staff Scientist in the Mathematical and Statistical Computing Laboratory. His research interests include biomedical image analysis and processing, in particular Diffusion Tensor MRI, use of statistical learning techniques in medicine, and application of complex network theory to neuroscience.

Christian Weimar is a consultant neurologist and an Associate Professor of Neurology at the University Hospital Essen, Germany. He received the MD Degree from the University of Freiburg, Germany, in 1995. He is the initiator and coordinator of the Center for Neurological Studies which comprises over 30 German academic and tertiary stroke care centres to conduct observational, prognostic studies in stroke patients. Over 25 peer reviewed papers have resulted from this collaboration. Other research interests include Alzheimer's disease, acute stroke treatment and stroke prevention.

Hans-Christoph Diener, MD, PhD is Professor of Neurology and Chairman of the Department of Neurology at the University of Duisburg-Essen, Germany. He is chairman of the West-German Headache Center, the Essen Pain Clinic and the outpatient rehabilitation unit (NETZ). He is the past President of the German Neurological Society, the President of the European Headache Federation and the President elect of the International Headache Society. He chairs the German Headache Consortium and the German Stroke Data Bank. His special research interests focus on headache, stroke and cerebellar physiology.

Andreas Ziegler is Full Professor and Head of the Institute of Medical Biometry and Statistics at the University at Lübeck. He received his PhD in Statistics in 1994 from the University of Dortmund, Germany. His research interests are related to methodological developments and applications in genetic and clinical epidemiology. He is Past President of the German Region of the International Biometric Society and currently serves on the editorial board of four journals. He has published over 200 refereed papers and five books. The latest one is 'A statistical approach to genetic epidemiology' and appeared with Wiley-VCH with coauthor Inke R. König.

1 Introduction

For a clinician, the most important questions a patient has usually are "What is the problem with me?", "What can you do about it?", and "Will I get better?". These three questions are related to the core themes of clinical epidemiology, namely, diagnosis, therapy, and prognosis, and methods to answer the last question will be the focus of this paper. However, informing the patient about the likely progress of his or her disease, i.e., patient-centered prognosis, is only one reason why prognostic studies may be important (Harrell et al., 1996). In a wider scope, the generation of accurate predictions in medical applications may help to better understand the underlying disease process, to inform clinical decisions, patients and family, as well as to define patient groups who are at a special risk (Altman and Lyman, 1998; Altman and Royston, 2000; Simon and Altman, 1994). Given this important role of biomedical prognosis, it comes as no surprise that a large body of literature on methods for developing multivariate predictors has been published. These include well-known regression approaches like the logistic regression. In addition, over the last decades, many new methods and combinations of them have been developed that can be utilised for anew tackling the problem of predicting patient outcomes.

All of these methods can be summarised under the term machine learning which has been nicely defined by Witten and Frank (2005). They consider "machine learning as the automated computer process of obtaining the best fitting model to a dataset". The chosen family of models is often highly parameterised and non-linear in the parameters, and the best fitting model is the one which minimises the chosen error criterion. As a special case, they defined statistical learning as machine learning in which the chosen error criterion reflects the distribution of errors of the data from the model.

As described below, interpretability of the model is an important criterion for the acceptance by clinicians. Therefore, many clinical epidemiologists still prefer the use of standard approaches. Nevertheless, the use of other machine learning algorithms has widely increased in the medical literature over the past decade at least in some selected areas. However, it seems that the broad familiarity with the newer approaches among medical statisticians is still wanting. The topic is, however, well-known to medical informaticians. An indication of this difference is that at the latest meeting of the International Medical Informatics Association 2004 in San Francisco, an entire session was devoted to "Machine Learning in Clinical Decision Making", and a tutorial on "Machine Learning Methods for Decision Support and Discovery" was offered. By comparison, nothing comparable was organised at the last Annual Conference of the International Society for Clinical Biostatistics 2006 in Geneva, Switzerland, or the XXIIIrd International Biometric Conference 2006 in Montreal, Canada.

The aim of this tutorial is to fill this gap and present an overview of some of the newer algorithms. For the application of a prognostic model, different criteria are relevant (see Section 6 for a detailed discussion). For example, in addition to the accuracy of the prognosis, the ease of application and computational time as well as the clinical interpretability may have to be considered. This already points to the fact that not one model will fit all applications and therefore we, by no means, aim at showing the superiority of any of the methods over any other. Instead, the focus will be on the description of methods that allow a patient-centered prognosis for binary outcomes in the medical setting. The practical use will be illustrated using data from a comprehensive prognostic factor study on the outcome of patients who have suffered from acute ischemic stroke and which has been published in great detail previously (Weimar et al., 2002, 2004; German Stroke Study Collaboration, 2004). The different algorithms will be evaluated with regard to practicability and clinical interpretability, and methods for systematically comparing prognostic methods will be presented. Because of space limitations for this tutorial, we will not discuss the problem of missing data in detail but assume data missing completely at random so that we eliminate incomplete case records from the data set. However, we comment on this topic in the discussion.

The mathematical sophistication of the several algorithms presented here varies considerably. Thus classification trees are perhaps easiest to understand, support vector machines probably hardest, while bagging, boosting and random forests are intermediate. This shifting level of difficulty is reflected in our presentation. The further questions of efficiency and optimality, for example large sample Bayes consistency is inevitably an active area of statistical research. We provide some guide to this literature, only, and do not attempt here a systematic survey or detailed analysis.

This tutorial is organised as follows: We begin by specifying the general framework of medical prognostic studies in Section 2. Section 3 describes the data set used for illustration. In this section, we also describe the logistic regression model (Section 3.5) which we used in the primary publications on the available data. We specifically chose the logistic regression model for comparisons because it is the best known and most frequently applied approach (Concato et al., 1993) in the biomedical literature for making yes/no predictions. It is considered the standard statistical learning machine and often used as benchmark for the comparison with new learning machines. Section 4 then introduces learning machines based on single classifiers, where Classification and Regression Trees (CART) (Breiman et al., 1984; Zhang and Singer, 1999) as well as Support Vector Machines (SVMs) (Cristianini and Shawe-Taylor, 2000; Noble, 2006; Schölkopf and Smola, 2002) will be considered. We chose these methods specifically because CART are often used as classifiers in ensembles, and because SVMs are becoming popular in a wide variety of biological applications. Section 5 presents the combination possibilities of single learning machines to ensembles. Here, the majority vote across different machines is usually taken for classifying subjects. In Section 6, we finally compare the different algorithms. Code pieces based on R software packages (*R* Development Core Team, 2005) are given along with the described methods.

2 General framework

A series of well-known papers (Altman and Lyman, 1998; Altman and Royston, 2000; Simon and Altman, 1994; Drew et al., 1999) has described reasons for developing prognostic models in medical applications, the important methodological challenges and guidelines for the conduct of medical prognostic studies. However, the principles of good study design, analysis and reporting are still less well appreciated for prognostic factor studies than for controlled clinical trials. For example, standards are available on the reporting of results from logistic regression analysis (Bagley et al., 2001) but, in contrast to the recommendations, the assumptions made in the analysis are not commonly reported (Ottenbacher et al., 2004). We therefore briefly summarise important principles for developing a prognostic model and sketch the different approaches for the validation of a prognostic model.

2.1 *Developing the prognostic model*

The first step to develop a prognostic model is to prospectively collect data from patients who are well defined by inclusion and exclusion criteria. In some situations, only a retrospective inclusion of the patients might be feasible; however, this may lead to more incomplete data sets and to biased assessments of the outcome (see, e.g., Sackett et al., 2000). After a sufficiently long and complete follow-up of these patients, the outcome is assessed, if possible blindly. This means that the clinician who evaluates the outcome is unaware of the patient's prognostic factors. In addition to the clinically relevant outcome and possible prognostic factors that are of specific interest in the respective study, all other important known prognostic factors have to be measured. However, although it is important that the model includes all relevant variables, at the same time it must not include more variables than are justified for a given sample size (Bagley et al., 2001). A reasonable standard rule of thumb is that the Events Per Variable (EPV) should be greater or equal to 10 (see, e.g., Simon and Altman, 1994; Bagley et al., 2001). For dichotomous outcomes, the number of events refers to the number of patients in the less frequent category. The required sample size does, however, not only depend on the number of variables. Instead, it depends on the study design, which in turn is determined by the aim of the study.

Some further specific issues include the question of how to adequately model a continuous variable (Royston et al., 2000; Royston and Sauerbrei, 2004) and the necessity for a stringent study protocol in advance of conducting the study (Simon and Altman, 1994; Drew et al., 1999).

After data collection, the resulting sample, comprising the *training data set* or *learning data set* is used to develop a prognostic model. The goal of this is to yield an algorithm that uses values from prognostic factors to accurately predict the patient's outcome. To evaluate the performance of the model, several measures can be applied (Harrell et al., 1996):

- *Predictive accuracy*: How well does the predicted event for an individual match the observed event?
- *Calibration*: How well does the predicted rate of events match the observed event rate?
- *Discrimination*: How well does the model distinguish between patients with different outcomes?

As a measure of predictive accuracy, we utilise the *concordance index*, also known as *hit rate*, which is defined as the proportion of patients in whom predictions and outcomes are concordant. Complementary to this, the *overall error fraction* is given by the proportion of patients with discordant prediction and observation. Concordance indices may also be calculated within certain patient groups instead of overall. If a binary outcome is considered, this results in the determination of *sensitivity* and *specificity* which estimate the probability that a patient belonging to one or the other outcome group is classified correctly. In addition, given estimated a priori probabilities for the outcome groups, positive and negative *predictive values* might be estimated.

2.2 Validating the prognostic model

After having thus obtained a satisfying prognostic model with high accuracy, the next question to answer is: How well does the model fare with patients collected from a different population than the original one? Different reasons for why this transfer might not succeed have been described, and Altman and Royston (2000) identified the following three:

- overoptimistic assessment of the predictive performance due to data-dependent aspects of model building
- weaknesses in study designs which can include no clear inclusion and exclusion criteria, missing data, or inadequate sample size
- models may not be transportable because of differences in sample compositions.

The first issue points to the fact that the data is utilised for both derivation of parameter estimates and testing model performance. Since the same data is used for both steps, the estimated performance of the model will often be overly optimistic, thus *overfit*. Flaws in the study design are difficult to address since these can only be corrected before any data is collected. However, even a careful study design does not guarantee too often that the prognostic model can be reliably transported to different data sets. Altman and Royston (2000) pointed out that different case-mix of the populations could lead to different model performances, if not all important prognostic variables are included in the model. Hence, as it will never be certain that every important variable is considered, it always has to be questioned whether the patients in the original sample were sufficiently similar to the patients for whom the prediction is made.

As a consequence of these problems, it has to be shown that a developed model performs reliably and sufficiently well in data sets different from the one it was developed. This means that the model must be *validated* before it can be utilised in practice. As a very general rule, this is achieved by making use of two data sets with the first being the training data in which the model is developed. The second data set, the *test data*, is then utilised to estimate the true predictive performance of the model. Depending on the specific data sets available, internal, external and temporal validation have to be distinguished. The different approaches for validating a prognostic model have been described in detail by Altman and Royston (2000); we have recently illustrated the different procedures using the data from the German Stroke Collaboration (König et al., 2007).

Internal validation uses only the training data to estimate the reproducibility of the model. This data set is artificially separated into disjoint data sets which then play the role of training and validation data. Approaches for this separation of the data set include sample splitting, k-fold Cross-Validation (CV), leave-one-out CV, leave-one-center-out CV and bootstrapping. Bootstrapping procedures (see, e.g., Hastie et al., 2003; Simon et al., 2003) are an integral part of some ensemble methods like bagging and will be described below.

The major difference between internal validation on the one hand and temporal and external validation on the other is that whereas internal validation as above utilises only one sample, for temporal and external validation additional independent data sets are appropriated, the test data. Estimating the error fraction in a *temporal validation* data set is similar to assessing the *reliability* of the model. That is, an estimate of the precision of the estimated predictive performance can be obtained. To be specific, the performance of the model is evaluated in new patients who are subsequently collected in the same study centers as before. Hence, the patients in this data set differ from the previous ones primarily with respect to a later time point of recruitment.

External validation is the most stringent validation of the clinical prognostic model. It is best suited to answer the question concerning the *validity* of the model, that is, whether the model actually evaluates what it is intended to evaluate. Here, it is required that the test data stem from patients who were recruited from different study centres than in the training data. An externally validated model therefore fares best with regard to generalisability, meaning robustness and broader usefulness of the results.

3 The data

Before describing different approaches to develop prognostic models, a short overview of the data used here is now given. The data were obtained from the German Stroke Data Bank funded by the Stiftung Deutsche Schlaganfall-Hilfe (German Stroke Foundation) and the German Stroke Study Collaboration funded by the Bundesministerium für Bildung und Forschung (German Ministry of Education and Research) in order to accurately predict the status of patients who had suffered from an acute ischemic stroke.

3.1 Endpoint

While the number of acute stroke trials to improve stroke treatment has increased considerably during the past decade, comprehensive knowledge about the impact of prognostic factors on outcome after acute ischemic stroke has been lacking because previous studies were mostly based on only exploratory analyses using small sample sizes. Of the many outcome parameters in use, guidelines acknowledge the importance of considering functional dependence and include impairments, disability and handicap experienced by the patient, in addition to mortality due to major bleeding complications (CPMP, 2000). The most commonly applied standardised outcome scale to assess functional dependence in stroke research is the Barthel index (BI) (CPMP, 2000; Mahoney and Barthel, 1965; Roberts and Counsell, 1998). It evaluates individual abilities for example in mobility or personal hygiene, that are immediately important to the patient and is considered an outcome being clinically relevant to the patient (Schuntermann, 1996). The BI can take values between 0 (total functional dependence) and 100 (total functional independence), and in practice, a suitable cut-off value is used to identify patients with complete restitution. We decided to use the BI assessed after 100 days as primary endpoint since a follow-up interval of 100 days represents the time span considered appropriate for therapeutic trials (CPMP, 2000). Furthermore, it is a trade-off to ensure that both rehabilitation is terminated and drop-out rate is kept low.

3.2 Possible prognostic factors

Prior to data assessment, we conducted a systematic literature review for independent prognostic factors for outcome after ischemic stroke in 1998. Based on these results and the clinical judgement on the impact of localisation of infarction, 49 variables assessable within the first 72 hours after admission were selected for further investigation. This time frame represents a compromise between a valid and accurate assessment of the variables of interest on one hand and an early prognosis on the other. Detailed information on the literature search, the variable selection process, and the final list of variables is available from the website http://www.uni-duisburg-essen.de/neurologie/stroke/free/lit_engl.html.

3.3 Training data set

Details on data assessment and management have been published previously (Weimar et al., 2002, 2004; König et al., 2003). In brief, data were collected in 1998 and 1999 prospectively in accordance to an extensive manual. All participating hospitals had an acute stroke unit. Seven of 23 centres (Essen, Benjamin Franklin Berlin, Frechen, Leipzig, Minden, München-Harlaching (only 1998), München-Großhadern (only 1998)) fulfilled pre-specified inclusion criteria by including more than 90% and following up more than 80% of the patients.

Ischemic stroke was defined as a focal neurological deficit of presumably vascular origin lasting >24 hours and excluding primary hemorrhage on initial cerebral imaging (Walker et al., 1981). In total, 1754 patients with ischemic stroke met the following criteria and were included in the analyses: No serious functional impairment (Rankin Scale <4) before the event to ensure that patients were functionally independent to a certain degree (CPMP, 2000), admission within 24 hours after stroke, not intubated during first 72 hours to allow for a valid assessment of all interesting variables, and survival in the first three days.

Data included all variables that had been identified in our literature search in addition to infarct localisation which had been chosen by clinical judgement. Of the 1754 patients, 1038 (59.2%) were men. Mean age of patients at baseline was 68.1 years (SD 12.7). The outcome was assessed on the BI within 80–150 (median 96) days after the event or by confirmation of death within 120 days after initial stroke. To identify patients with complete restitution as advocated for clinical trials (CPMP, 2000), a cut-off value of BI $\geq$ 95 (complete restitution) vs. <95 (incomplete restitution or mortality) was used. 1025 patients (58.4%) were completely restituted, 563 patients (32.1%) were incompletely restituted, and 166 patients (9.5%) had died. One hundred and eight patients (6.1%) had received thrombolysis. Complete data was available in 1737 (99.0%) of the patients.

All patients gave informed consent if their personal data were to be transferred to the data management centre. Aspects of data safety of the Stroke Database were considered to be clarified by the responsible data protection officer.

3.4 Test data set

Enrollment of patients started on 1st, February, 2001, and was terminated on 15th, March 2002, after the calculated required number of patients had been reached (Weimar et al., 2002, 2004; König et al., 2003). Four neurologic departments participating in the initial study also took part in the validation study (Essen, München-Harlaching, München-Großhadern, Minden). These allow a temporal validation (König et al., 2007). Nine neurologic departments participated in the validation study only (Charité Berlin, Bielefeld, Bonn, Jena, Magdeburg, Rostock, Saarland, Stuttgart, Ulm) and represent centres for external validation (König et al., 2007).

All participating hospitals operate an acute stroke unit. On admission, the treating physician reported the admission of every stroke patient via fax to the coordinating centre. According to the study protocol (König et al., 2003), we excluded all patients from those centres with less than 75% follow-up or a drop-out rate of >10%. The drop-out rate was defined as the proportion of initially reported patients who could not be included in the validation study because of missing baseline information. The remaining patients were included if they met inclusion criteria similar to those of the initial study (German Stroke Study Collaboration, 2004).

In the test data, a subset of variables assessed in the training phase was included. In total, 38 predictive variables shown in Appendix A that had been assessed previously were also available in the test data set.

We included only those patients with complete baseline and follow-up information obtained between 85 and 120 days after admission. Details on loss to follow up can be found elsewhere (German Stroke Study Collaboration, 2004). Of the 1470 patients with complete follow-up, 57.3% were men. Mean age of patients at baseline was 67.9 years (SD 12.4). After 100 days, 831 patients (56.5%) were completely restituted ($\mathrm{BI} \geq 95$), 526 patients (35.8%) were incompletely restituted ($\mathrm{BI} < 95$), and 113 patients (7.7%) had died. Complete case records were available in 1447 (98.4%) patients.

Patients were informed about study participation and informed written consent was obtained to forward personal data to the coordinating center. Patients were treated according to best current knowledge in clinical routine. The study was approved by the Ethics Committee of the University of Essen and aspects of data safety were approved by the responsible data protection state representative.

3.5 Logistic regression

As primary analysis, a prognostic model was developed in the training data using the logistic regression model to predict complete functional restitution vs. incomplete restitution or mortality.

In the first step, descriptive statistics were obtained for all 49 variables and the recruiting centre. Because of a frequency of less than 4% in one of the alternative categories, four binary variables were excluded (localisation of the infarct in the anterior cerebral artery, borderline middle/posterior cerebral arteries, borderline anterior/middle cerebral arteries, and long perforating arteries). Another three single variables were eliminated because of substantive correlations with other variables and less predictive value or reliability than the respective correlated variable (National Institute of Health-Stroke Scale (NIH-SS), items left leg paresis, right leg paresis, and commands).

To model the relationship between continuous variables (age and body mass index) and outcome, univariate fractional polynomials were used on randomly selected 25% of the total sample (Royston et al., 2000). For age, the best fit was obtained including only the linear term. No significant gain was achieved by including body mass index; this variable was therefore excluded. The ordinal variables NIH-SS total score and Rankin Scale were treated as linear variables in the regression models. We used this method because a linear fit is regarded as the natural approach when expecting a monotone effect of the ordinal covariate (Agresti, 1990). In addition, we applied fractional polynomials on randomly selected 25% of the total sample to model the relationship between these two scores and the outcome. For both variables, the best fit was obtained including only the linear term. In addition, no significant center effect was observed prior to multivariate model building upon use of logistic regression with dummy-coded center variables (likelihood ratio test $p > 0.1$).

The remaining 38 variables were fitted using logistic regression models via forward, backward and stepwise selection. The EPV was >20. Nevertheless, variables were retained only if their resulting p-value was $\geq$0.005 (Altman and Lyman, 1998). From models with all variables that resulted from any of the selection procedures, any variable with $p > 0.005$ was eliminated backward. To the remaining set of variables, every previously eliminated variable was again added and kept in the model if it fulfilled the same criteria. Finally, all two-way interactions of the resulting variables were investigated and kept if $p < 0.005$ and if, for categorical variables, each cell contained at least 4% of observations.

We are fully aware that our strategy for exclusion of variables and variable selection is risky and need not be reliable. It is simple to construct examples where variables x_1 and x_2 are highly correlated with each other, both by themselves show no correlation with the outcome variable, but when considered jointly in the model lead to excellent prediction.

Table 1 Logistic regression model in the training data

Variable	$\hat{\beta}$	$S.E._{\hat{\beta}}$	OR	95% CI
Intercept	−8.378	0.510		
Neurological complications	1.289	0.332	3.628	1.892–6.958
Fever > 38°	1.078	0.238	2.937	1.842–4.683
Lenticulostriate arteries infarction	0.743	0.216	2.103	1.378–3.209
Diabetes mellitus	0.655	0.152	1.925	1.430–2.591
Rankin scale (Difference of 1 scale score)	0.537	0.070	1.710	1.490–1.963
Prior stroke	0.517	0.161	1.676	1.223–2.297
Left arm paresis (Difference of 1 scale score)	0.488	0.103	1.629	1.330–1.994
Female gender	0.394	0.138	1.484	1.131–1.946
Right arm paresis (Difference of 1 scale score)	0.393	0.089	1.481	1.244–1.763
NIH-SS total score at admission (Difference of 1 scale score)	0.074	0.024	1.077	1.028–1.129
Age (Difference of 1 year)	0.066	0.006	1.068	1.055–1.082

$\hat{\beta}$, parameter estimate; S.E.: Standard Error; OR: Odds Ratio; CI: Confidence Interval; NIH-SS: National Institute of Health-Stroke Scale.

For the final models, parameter estimates with standard errors (S.E.), odds ratios (OR) and asymptotic 95% confidence intervals (CI) for all variables were calculated and are presented in Table 1. The model explained $R^2 = 61.7\%$ of the complete variation according to McKelvey and Zavoina (1975). Leave-one-out cross-validation was used to estimate the shrinkage factor γ (Verweij and Van Houwelingen, 1993): A factor of $\gamma = 0.973$ was obtained. The threshold for classification using the logistic distribution function was set to 0.437 so that the predicted proportion of events was equal to the observed. The percentage of thus correctly classified patients was estimated (Verweij and Van Houwelingen, 1993). In addition, we defined specificity as the fraction of correctly classified patients who were completely restituted after three months, and sensitivity as the respective fraction of correctly classified patients who were only incompletely restituted or deceased (see Table 3).

We employed a ten-fold cross-validation for internally validating the model. To be specific, we performed the entire model development as described above in every single one of the ten splits. The cross-validated error fractions were then calculated as the average of the ten single error fractions as shown in Table 3.

For temporal validation, data of all 581 patients in the test data set were utilised who had been recruited in the same study centres as the patients in the training data. External validation of the model was based on the data of all 866 patients coming from different study centres. Using the shrunken estimates and the predefined threshold of the previously developed model, patients in both subsets were classified. The quality of classification in terms of correctly classified patients was determined. Table 3 depicts the hit rate (overall accuracy fraction) as well as sensitivity and specificity. To compare accuracy fractions between internal, temporal, and external data sets, we computed 95% confidence intervals on the difference of proportions according to (Newcombe, 1998) (see Section 6.1) as well as two-sided exact p-values from Fisher's exact test. Correct classification fractions do not differ substantially for internal and either temporal or external validation (difference $= -0.0198$ [-0.0554; 0.0189] and 0.0276 [-0.0058; 0.0625], $p = 0.34$ and 0.11, respectively). However, classification results are significantly worse for the external data compared with the temporal validation data (0.0474 [0.0040; 0.0893], $p = 0.03$).

4 Learning machines: single classifiers

Having outlined the general framework, our example data and then our primary analysis, we now turn to the description of the typical scenario for the use of learning machines. In the setting of a medical prognostic study, we have a set of n patients who have experienced a clinically relevant dichotomous outcome y_i. Furthermore, for every patient we have collected data on a set of p predictive variables $x_i = (x_{i1}, \ldots, x_{ip})'$, called the *features*. The aim of the prognosis is to predict the dichotomous outcome based on the predictive variables, and we thus have a *classification* problem. This problem is also termed a *supervised learning* problem. This means that, for every patient, values of both predictive variables and the outcome are known. Using an algorithm, a prediction $\hat{y}_i$ is generated and can be compared with the 'supervisor' y_i.

In *unsupervised learning*, only values of predictive variables are available. Here, the aim is to infer associations or structures among the variables and to identify useful low-dimensional spaces within the original space. Examples for these approaches are principal components, multidimensional scaling, cluster analysis and self-organising maps (Hastie et al., 2003). As a comparison with the real answer is impossible, it is generally difficult to ascertain the validity of inferences drawn from the result of unsupervised learning machines. This is a somewhat controversial area still under active research.

In medical prognostic studies, a central goal is not necessarily to detect underlying patterns in the available data to derive or uncover an inherent structure. Instead, the collection of patients with values in both predictive and outcome variables is essential. In some applications, it may also be of subsequent interest to identify subsets of patients who are *at increased risk*. In this paper, we therefore focus on supervised learning machines only, and we begin with stand-alone learning machines.

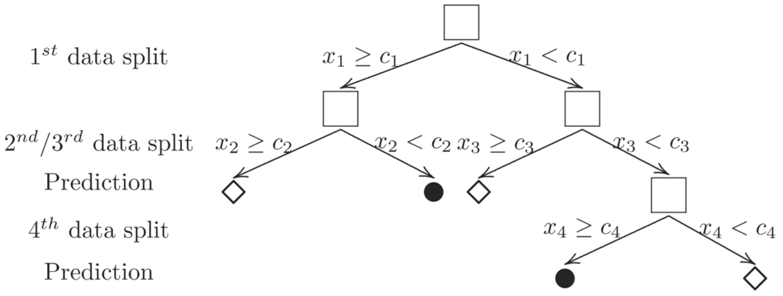
We start with CART (Section 4.1) which were developed by Breiman et al. (1984). Over the past 20 years, they have been widely used in different areas including computer science, biology, psychology, and prognostic and diagnostic clinical studies (e.g., Costanza and Paccaud, 2004; Schwarzer et al., 2003; Smolle and Gerger, 2003). They are implemented in many freely and commercially available software packages. In the 1990s, SVMs that belong to the family of so-called kernel methods were developed (Cristianini and Shawe-Taylor, 2000; Noble, 2006; Schölkopf and Smola, 2002). They have been employed in a variety of application fields, including biomedical research, and here they are primarily used for the analysis of gene expression data (see, e.g., Simon et al., 2003). Different easy-to-use implementations are freely available. They have, however, been rarely applied in classical biomedical studies like patient-centered prognosis.

4.1 Classification and regression trees

The overall goal of CART is to generate a *decision tree* that classifies patients correctly. Beginning at the *stem* or *root node* with the entire sample of patients, one follows the stem into its branches. At each node of the tree, a split in the sample is made, until, in the last branches, only a relatively homogeneous subset of patients remain.

Technically speaking, the general idea of CART is to determine a sequence of logical 'if-then' conditions that permit an accurate classification of cases (see Figure 1). Beginning with the entire data as the first node, the feature space is partitioned into two branches. These in turn become the nodes for the next partitioning. Hence, the procedure is described as a binary recursive partitioning algorithm; binary, because a parent node is always split into exactly two child nodes, and recursive, because the process can be repeated by treating each child node as a parent. The objective in the partitioning is to identify subgroups of patients who are increasingly homogeneous with respect to their outcome.

Figure 1 Dropping a case down a classification tree. Four data splits c_1, c_2, c_3, and c_4 are shown and cases are classified according to their feature values x_1, x_2, x_3, and x_4. Bullets represent the predicted outcomes $\hat{y}_i = +1$, while predicted outcomes $\hat{y}_i = -1$ are depicted by diamonds



The Classification and Regression Tree (CART) procedure

The CART procedure may be separated into the following four steps which are described below in more detail:

1. A tree is grown.
2. The tree growing process is stopped.
3. Branches of the tree are pruned.
4. The optimal tree is selected.

1 Growing the tree

To grow a classification tree, these steps are performed:

a One feature x_j with $j = 1, \ldots, p$ out of the entire feature space is selected.

b For each possible data split, also termed partition, using this x_j, the observations in both partitions are classified. Usually, each new case is assigned to the majority class in this partition.

c Considering all possible data splits within every possible feature, the best split c_1 is selected that minimises the impurity in the resulting partitions. Since all splits using every feature are searched, there is a large number of candidate splits, and usually a brute force search through all splits is conducted.

d Within one of the resulting partitions, again one feature x_j is selected.

e Considering all possible data splits in this partition, the best split c_2 is selected that minimises the error within the respective partition.

f Steps (d) and (e) are repeated.

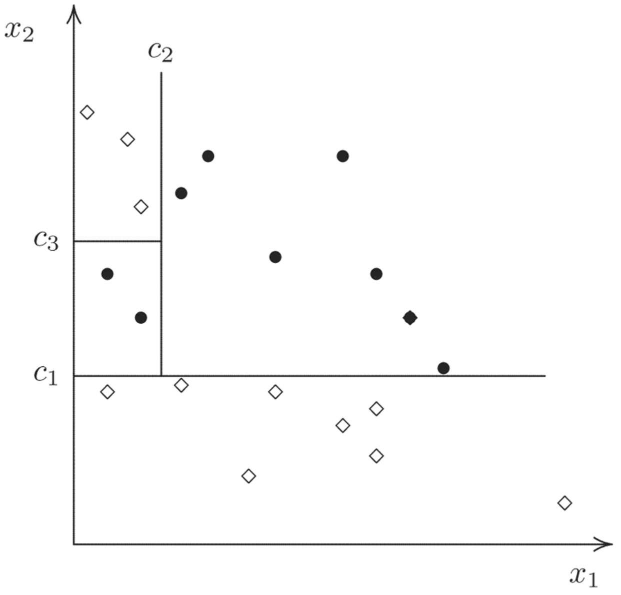
Figure 2 illustrates the results of this algorithm with two features x_1 and x_2. Upon use of these data splits, a classification tree with the splits depicted as nodes can be built as displayed in Figure 1. Dropping a case down the tree results in the classification of the patient. The simplest version of a tree contains only two nodes, resulting from a single data split, and is called a *stump*.

In choosing the best partitioning, CART seeks to maximise the average purity of the two child nodes. Different measures of purity, i.e., splitting criteria can be applied. With p being the proportion of observations in one of the two outcome classes, we define:

Misclassification error: $1 - \max(p, 1 - p)$,
Gini index: $2p\,(1 - p)$,
Deviance: $-p \cdot \ln(p) - (1 - p) \cdot \ln(1 - p)$.

The misclassification error is the typical one to use, although the Gini index and the deviance are more sensitive to changes in the node probabilities. In our applications we prefer the use of the Gini index because it is related to the variance. More splitting rules and arguments for choosing a specific splitting rule are given in Zhang and Singer (1999). Instead of minimising the impurity as pointed out in (c), the aim can be defined more generally to minimise costs. This distinction becomes important when some predictions that fail have more severe consequences than others, or when some predictions that fail occur more frequently than others. To this end, different losses can be assigned by weighting the observations in each class differently. Hence, the prior probabilities in the classes are changed which then influence the growing of the tree.

Figure 2 Separation of patients by a classification tree. Three data splits c_1, c_2 and c_3 are shown for features x_1 and x_2. Bullets represent $y_i = +1$, while $y_i = -1$ is depicted by diamonds



2 Stopping the tree building

One important parameter for growing trees is the size of the trees, or equivalently, the stopping rule that is applied. Larger trees have the advantage of exploiting more of the available information for accurate predictions. However, larger trees typically lead to overfitting the data, with subsequent loss in generalisation to new data. With larger trees the results are more difficult to interpret. Smaller trees, on the other hand, are easier to understand but might not adequately reflect complex data structures. Common stopping rules are: stop the growing process when

1 only cases with the same outcome remain in every child node

2 all cases within every child node have identical predictor variables

3 an external limit on the *depth* or complexity of the tree, i.e., the number of levels in the tree has been reached.

It has, however, been pointed out that it is usually impossible to specify a reliable stopping rule in advance. Instead, *maximal* trees are grown and then pruned in order to achieve optimal sized trees.

3 Pruning the tree

Following Breiman et al. (1984), pruning a tree starts from a tree grown to its maximal size. Beginning at the terminal nodes, child nodes are subsequently cut (collapsed upwards), i.e., *pruned* away from the branch. This process depends on the definition of a complexity parameter ζ that governs the tradeoff between complexity, i.e., the number of nodes of the tree and its accuracy for future data. Pruning is continued as long as the resulting change in misclassification cost is less than ζ times the change in tree complexity; thus, ζ is a measure of how much additional accuracy a split must add to the entire tree to warrant the additional complexity (Breiman et al., 1984). With different ζ, different 'optimal' trees are obtained.

4 Selecting the optimal tree

The final task consists of selecting the most accurate tree from the set of pruned trees. If no independent data set is available to better estimate the accuracy, internal validation is used which is based on the original training data set.

Although the procedure of growing classification trees is intuitive, there are some disadvantages to CART. These include the problem that there may be competing splits at some nodes which result in the same purity (see Figure 3(a)). The usefulness of the resulting trees will suffer from the fact that the trees have a high variance in many situations. That is to say, small changes in the data may result in extremely different trees, thus different interpretations, distinct predictions for individual cases and widely varying error fractions. Also, when the theoretical and distributional assumptions of more traditional methods are met, then these may be preferable; a comparative discussion of CART with logistic regression has been given in Zhang and Singer (1999).

Finally, there may be sample data structures for which the basic tree algorithm is ineffective. Three such examples are presented in Figure 3. In Figure 3(a) we see that any initial split, on variable x_1 or on variable x_2, does not produce a useful classification. However, it is also true that subsequent splits on x_1 and x_2 yield accurate classification. In Figure 3(b) and (c) we again see that no useful split arises when using just x_1 or x_2. However, efficient splits are possible with appropriate modifications of the explanatory variables. In the first case, displayed in Figure 3(a), two branches of a hyperbola have to be added, while linear and quadratic functions need to be modelled in the second and third example, respectively. The implication is here that inclusion of elementary functions of the data may effectively handle quite diverse data structures. A nice example of this is developed in the section on feature space and kernel functions using the kernel function technology deployed in support vector machines; more on this is presented below (see Section 4.2).

Example for CART

To grow a classification tree `method = "class"` on the training data from the stroke patients, we utilised the *R*-package *rpart* by Therneau and Atkinson, maintained by Ripley. *R* (*R* Development Core Team, 2005) has a home page at http://www.R-project.org and is freely distributed under a GNU-style copyleft. We specifically employed the Gini index `parms = list(split = "gini")` as split criterion in tree growth (step 1). The stopping of the tree growth (step 2) was accelerated by restricting the minimum number of observations in a node to 20 (`minsplit = 20`). In step 3, the tree was pruned by setting the complexity parameter to 0.01 (cp = 0.01). As a result, splits that decrease the lack of fit by a factor of less than 0.01 were not conducted. Hence, whereas the original tree utilised 22 of the 38 available variables, only three were kept in the pruned tree. The resulting tree is displayed in a flow chart in Figure 4, and the *R*-code for tree building is

```
tree.train <− rpart(y  ~., data = train, method = "class",
                    parms = list(split = "gini"),
                    control = rpart.control
                    (minsplit = 20, cp = 0.01))
```

Here, ~ separates the outcome train$y from the features in the training data. The 2×2 table of prediction in the training data is obtained by:

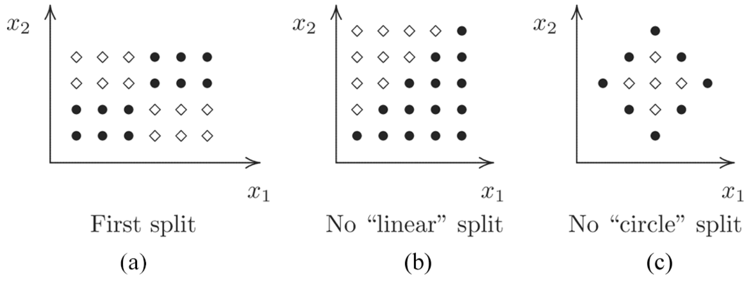
```
table(predict(tree.train, type = "class"), train$y)
```

The classification tree thus obtained was then applied to the training data to compute error fractions, sensitivity, and specificity. Predictions for the validation data are created using the object tree.train from the training data with the features from the test data by

```
tree.test <− predict(tree.train, newdata = test, type = "class")
```

We also employed ten-fold cross-validation on the training data to estimate the respective internally validated values. The results are displayed in Table 3. Temporal and external validation were performed by applying the fixed classification tree to the temporal and external subsets of the test data sample (see Table 3).

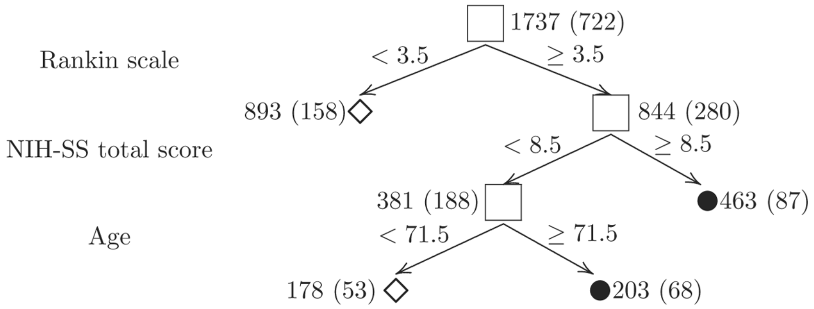
Figure 3 Problematic data sets for classification trees: No reasonable first split can be made in the figure on the left. Classification trees may not allow linear splits, which would be reasonable in the figure displayed in the middle. Similarly, a circle can separate the two groups in the figure on the right. However, this may not be possible with a given classification tree algorithm, unless appropriate functions are included in the list of explanatory variables



4.2 Support vector machines

SVMs require more mathematics than the other approaches described here. We aim at minimising the in-text mathematics in order to present the intuitive ideas of this method. Therefore, we confined mathematical details to Appendix B.

Figure 4 Classification tree for stroke data. Diamonds represent a predicted complete restitution, while predicted incomplete restitution or mortality is depicted by bullets. Number of patients (loss values) are given



A starting point is the classification problem as depicted in Figure 5. For each outcome $y_i = +1$ (solid bullets) or $y_i = -1$ (open diamonds) we have an associated vector of explanatory variables $\boldsymbol{x}_i \in \mathbb{R}^p$. The aim is to find a hyperplane $f(\boldsymbol{x})$, a straight line in Figure 5, which best separates the groups. The equation for a hyperplane is given by (see Appendix B)

$$\{\boldsymbol{x} \in \mathbb{R}^p : f(\boldsymbol{x}) = \boldsymbol{x}'\boldsymbol{\beta} + \beta_0 = 0\}.$$

Multiplication of both $\boldsymbol{x}'\boldsymbol{\beta}$ and β_0 with the same non-zero constant results in the same hyperplane. Therefore, we restrict $\boldsymbol{\beta}$ to the unit vector, i.e. $\|\boldsymbol{\beta}\| = 1$.

The separable case

We first consider the case of separable classes, where by definition we can find a function $f(\boldsymbol{x}) = \boldsymbol{x}'\boldsymbol{\beta} + \beta_0$ such that $y_i f(\boldsymbol{x}_i) > 0$ for all $i = 1, \ldots, n$. Obviously, there is an infinite number of separating hyperplanes for the problem displayed in Figure 5. To uniquely determine a separating hyperplane under the SVM criteria, we therefore use the hyperplane which has the biggest *margin* between the classes (Figure 6). The band is C units away from the hyperplane on either side, and the margin thus is $2 \cdot C$ units wide.

The maximisation problem can therefore be expressed as

$$\max_{\boldsymbol{\beta}, \beta_0, \|\boldsymbol{\beta}\|=1} C \quad \text{subject to} \quad y_i\big(\boldsymbol{x}_i'\boldsymbol{\beta} + \beta_0\big) \geq C, \quad i = 1, \ldots, n. \tag{1}$$

This optimisation problem is restated in Appendix B.3, and its solution is outlined in Appendix B.4. Once the separating hyperplane $f(\boldsymbol{x})$ has been determined, the classification rule for an observation $\boldsymbol{x}_*$ of explanatory variables can be written as

$$\operatorname{sign}\big(f(\boldsymbol{x}_*)\big) = \operatorname{sign}\big(\boldsymbol{x}_*'\boldsymbol{\beta} + \beta_0\big). \tag{2}$$

This means that observations $\boldsymbol{x}_*$ are classified either as $+1$ or -1 depending on the side on which they fall on the hyperplane.

Figure 5 A separable classification problem together with the separating hyperplane $\{x : x'\beta + \beta_0 = 0\}$. The hyperplane represents the decision boundary. Solid bullets represent $y_i = +1$, while $y_i = -1$ is depicted by open diamonds. The two-dimensional space represents the vector space of the features x

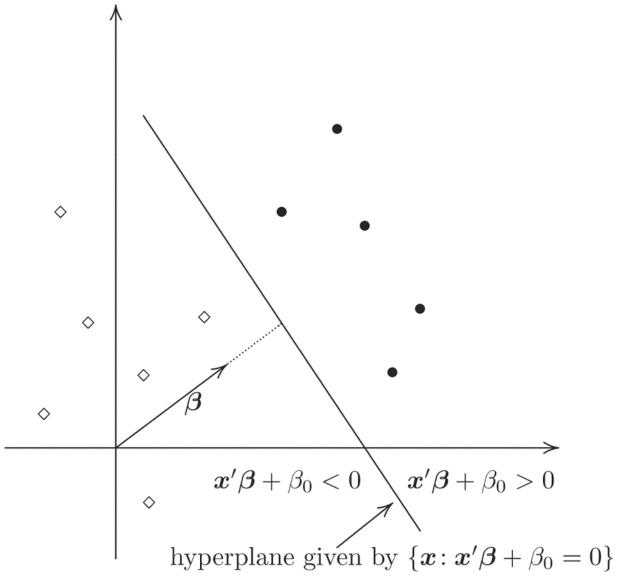
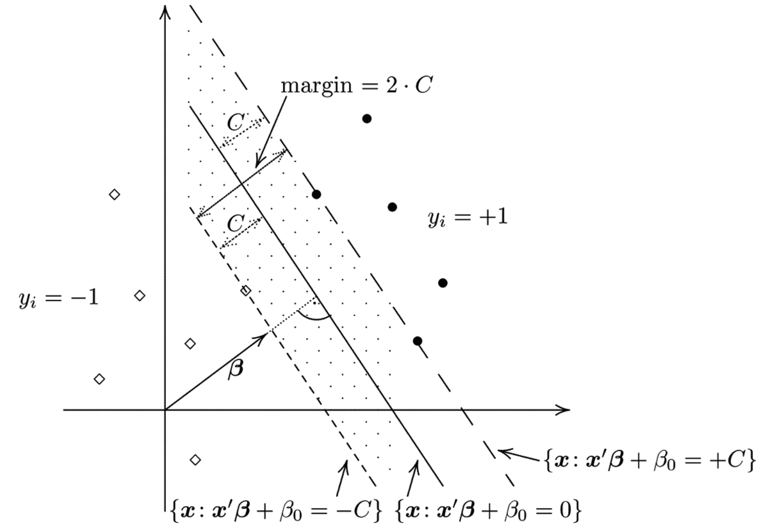

$\boldsymbol{\beta}$
$\boldsymbol{x}'\boldsymbol{\beta} + \beta_0 < 0$
$\boldsymbol{x}'\boldsymbol{\beta} + \beta_0 > 0$
hyperplane given by $\{\boldsymbol{x} \colon \boldsymbol{x}'\boldsymbol{\beta} + \beta_0 = 0\}$

Figure 6 The separable classification problem together with the separating hyperplane and the margin. The band is C units away from the hyperplane on either side. Thus the margin is $2 \cdot C$ wide


margin $= 2 \cdot C$
C
C
$y_i = +1$
$y_i = -1$
$\boldsymbol{\beta}$
$\{\boldsymbol{x} \colon \boldsymbol{x}'\boldsymbol{\beta} + \beta_0 = +C\}$
$\{\boldsymbol{x} \colon \boldsymbol{x}'\boldsymbol{\beta} + \beta_0 = -C\}$
$\{\boldsymbol{x} \colon \boldsymbol{x}'\boldsymbol{\beta} + \beta_0 = 0\}$

The non-separable case

We next assume overlapping classes so that they cannot be separated perfectly by a hyperplane (Figure 7). The aim, however, still is the maximisation of the margin $2 \cdot C$

though we now have to allow for some points falling within the margin, and also some points falling on the wrong side of its margin. In these cases, the condition

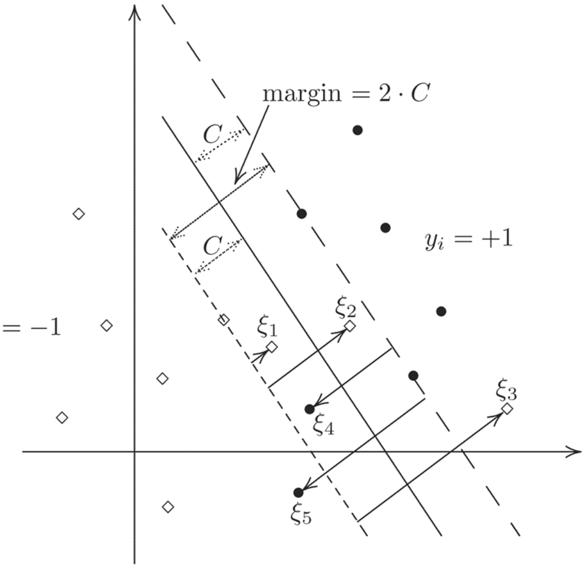
$$y_i f(\boldsymbol{x}_i) = y_i\big(\boldsymbol{x}_i'\boldsymbol{\beta} + \beta_0\big) \geq C$$

cannot be fulfilled for all observations i. Instead, the distance will be smaller for some observations i by a margin factor $(1 - \xi_i)$, where $\xi_i \geq 0$ for all i, i.e.,

$$y_i f(\boldsymbol{x}_i) = y_i\big(\boldsymbol{x}_i'\boldsymbol{\beta} + \beta_0\big) \geq C(1 - \xi_i).$$

For a point that falls on the correct side of its margin, ξ_i still is 0. Note that though $\xi_i > 0$ if a point falls on the wrong side of its margin, a misclassification only occurs if $\xi_i > 1$. If the sum $\sum_{i=1}^{n} \xi_i$ is bounded from above by a constant γ, a user-defined *tuning parameter*, then the total proportional amount by which predictions $f(\boldsymbol{x}_i)$ are on the wrong of their margin is also bounded. The margin grows inversely with γ, i.e., it is larger for small γ, and smaller for large γ. The slack variables ξ_i can thus be interpreted as the proportional amount by which the prediction $f(\boldsymbol{x}_i)$ is on the wrong side of its margin.

Figure 7 The non-separable classification problem together with the separating hyperplane and the margin. The points labelled ξ_i are said to be on the *wrong side of their margin*. Points on the *correct side* have $\xi_i = 0$



We aim at maximising the margin C subject to $\|\boldsymbol{\beta}\| = 1$ as in the separable case. However, we now have an additional boundary condition $\sum_{i=1}^{n} \xi_i \leq \gamma$. This leads to a quadratic optimisation problem with linear constraints, and its solution can thus be obtained by a quadratic programming algorithm using Lagrange multipliers. Details are described in Appendix C. In the following and in Appendix B.4, α_i denote the Lagrangian multipliers for the constraints

$$y_i\big(\boldsymbol{x}_i'\boldsymbol{\beta} + \beta_0\big) \geq 1 - \xi_i. \tag{3}$$

It is sketched in Appendix B.4 that the minimisation problem has a solution with the simple form

$$\hat{\boldsymbol{\beta}} = \sum_{i=1}^{n} \hat{\alpha}_i y_i \boldsymbol{x}_i. \tag{4}$$

Importantly, the coefficients $\hat{\alpha}_i$ are only greater 0 for those observations i for which the equality sign holds in constraint (3). These observations are the *support vectors* because $\hat{\boldsymbol{\beta}}$ is represented in terms of these alone. Note that only these observations are critical for the classification problem. Hence, an observation on the correct side and far away from its margin, will not contribute to the classification problem. Only those observations will become support vectors that are either on the wrong side or close to the boundary of their margin. The existence and nature of support vectors in the SVM approach thus strongly distinguishes this method from a classical linear discriminant approach, to which SVM might seem outwardly similar. In effect, the SVM only 'sees' the data near the decision boundary. Nonetheless, despite this important mathematical difference a linear SVM using the original input data to specify the hyperplane decision boundary, may generate a classifier that is almost identical to a linear discriminant. That the SVM approach has considerably more theoretical richness, and thus more practical flexibility, is revealed in the next section wherein a systematic method is discussed that provides for new features, while still maintaining a linear outlook using hyperplanes.

Feature space and kernel functions

Figure 8(a) illustrates that a good separation by a single cut may be impossible. Here, the linear space is considered with $y_i = +1$ at the tails of the distribution of the x-variable. If, however, the dimension of the feature space is increased by a so-called kernel function, the SVM might be able to find a good hyperplane decision boundary in some enlarged and transformed space of features (see Figure 8(b)).

To this end, we consider transformations of the data, $\boldsymbol{h}(\boldsymbol{x})$, into a higher-dimensional *feature space* so that the separating hyperplane can be written as

$$\boldsymbol{h}(\boldsymbol{x})'\boldsymbol{\beta} + \beta_0.$$

The disadvantage of this increase in dimensionality are the CPU–time expensive calculations, but this computational effort may be reduced significantly by using a so-called kernel function which we now describe.

To start, we combine the decision rule (2) and the solution of the optimisation problem (4). This leads to

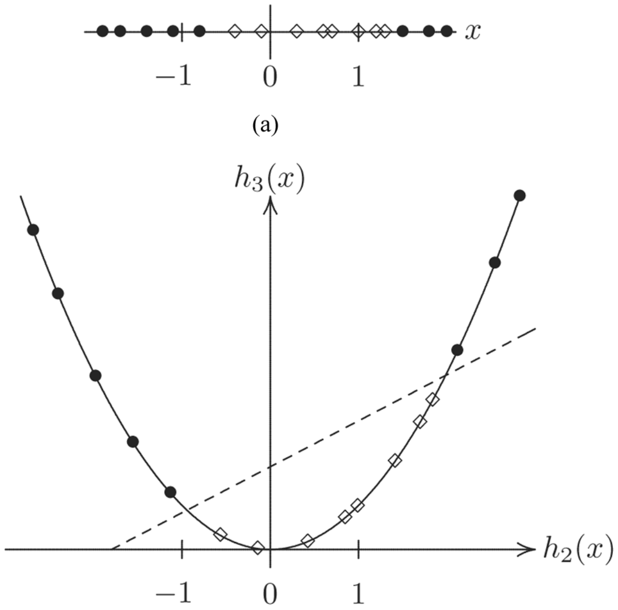
$$\operatorname{sign}\big(f(\boldsymbol{x}_*)\big) = \operatorname{sign}\left(\sum_{i=1}^{n} \alpha_i y_i \boldsymbol{x}_*' \boldsymbol{x}_i + \beta_0\right) = \operatorname{sign}\left(\sum_{i=1}^{n} \alpha_i y_i \langle \boldsymbol{x}_*, \boldsymbol{x}_i \rangle + \beta_0\right) \tag{5}$$

for true Lagrangian multipliers α_i and the usual inner product $\langle \cdot, \cdot \rangle$.

If the feature space is used instead of the input space, equation (5) is replaced by

$$\operatorname{sign}\big(f(\boldsymbol{x}_*)\big) = \operatorname{sign}\left(\sum_{i=1}^{n} \alpha_i y_i \langle \boldsymbol{h}(\boldsymbol{x}_*), \boldsymbol{h}(\boldsymbol{x}_i) \rangle + \beta_0\right). \tag{6}$$

Figure 8 Example of (a) a one-dimensional input space and (b) a higher dimensional feature space. No single cut separates bullets and diamonds in the linear space (a). The feature space (b) allows separation by a hyperplane. The functions h_2 and h_3 are two components of the inhomogeneous quadratic kernel with $c = 1$ from equation (7); the univariate input x from (a) is transformed via this kernel to the two-dimensional feature space (b). Perfect separation is possible with the displayed data in this feature space



(a)

(b)

Calculations may be reduced drastically by using a function K, such that

$$K(\boldsymbol{x}, \boldsymbol{x}_i) = \langle \boldsymbol{h}(\boldsymbol{x}), \boldsymbol{h}(\boldsymbol{x}_i) \rangle$$

allows the computation of the value of the inner product without recurrent explicit computation of the function h. The function K is called the *kernel function* and should be symmetric and positive definite. On a more abstract level, Reproducing Kernel Hilbert Spaces (RKHS) are used for kernel definitions so that the Riesz representation theorem may be applied (Reed and Simon, 1980). Properties of kernel functions and of RKHS are discussed in detail, e.g., in Schölkopf and Smola (2002).

Popular choices for kernel functions are sigmoid kernels, also termed neural network kernels, $K(\boldsymbol{x}, \boldsymbol{x}_i) = \tanh(\kappa \langle \boldsymbol{x}, \boldsymbol{x}_i \rangle + \vartheta)$, where $\kappa > 0$ and $\vartheta < 0$, Gaussian radial basis function kernels $K(\boldsymbol{x}, \boldsymbol{x}_i) = \exp(-\frac{1}{2\sigma^2} \|\boldsymbol{x} - \boldsymbol{x}_i\|^2)$, with $\sigma^2 > 0$, and, finally, inhomogeneous dth degree polynomial kernels, where $d \in \mathbb{N}$ and $c \geq 0$: $K(\boldsymbol{x}, \boldsymbol{x}_i) = (c + \langle \boldsymbol{x}, \boldsymbol{x}_i \rangle)^d$.

In the example of Figure 8, we used an inhomogeneous quadratic kernel $K(\boldsymbol{x}, \boldsymbol{x}_i) = (1 + \langle \boldsymbol{x}, \boldsymbol{x}_i \rangle)^2$ with $c = 1$ and univariate input so that $\boldsymbol{x} = x$ and $\boldsymbol{x}_i = x_i$.

This kernel yields $\boldsymbol{h}(x) = \big(h_1(x), h_2(x), h_3(x)\big) = \big(1, \sqrt{2}x, x^2\big)$ since

$$K(x, x_i) = \langle \boldsymbol{h}(x), \boldsymbol{h}(x_i)\rangle = \big(1 + \langle x, x_i\rangle\big)^2 = 1 + (\sqrt{2}x)(\sqrt{2}x_i) + x^2 x_i^2. \tag{7}$$

For illustration, we displayed only h_2 and h_3 in Figure 8 because $h_1(x) = 1$.

Example for SVM

We utilised the R-package *e1071* to train and test SVMs in the stroke data sets. To be specific, SVMs with a sigmoid, radial, and polynomial kernel function were developed on the training data set. To select suitable function parameters, a grid search over possible parameter values ($\texttt{gamma} = 0.005 + 0.005 * (0:2)$) was employed using the tuning function. As an additional parameter, the cost of a constraints violation was tuned ($\texttt{cost1} + 0.05 * (0:10)$). It should be noted that this tuning might require large CPU–time; for instance, conducting a grid search over 3 times 11 parameter values already took 10 minutes on a Pentium IV with 2.8 GHz. The code piece for tuning the SVM in the training data with a radial kernel is

```
svm.rad.tune <- tune.svm(train.x, train.y, kernel = "radial",
                         gamma = 0.005 + 0.005*(0 : 2),
                         cost = 1 + 0.05*(0 : 10))
```

Here, `train.x` is a data matrix with features in the training data, and `train.y` is the factor of outcomes in the training data. The performance of the SVMs for each constellation can be displayed by `svm.rad.tune$performance`. After tuning, an SVM with a specific parameter constellation may be computed:

```
svm.train.rad <-svm(train.x, train.y, kernel = "radial", gamma = 0.01,
                    cost = 1.4)
```

The next two commands show the predicted values for the training data and the 2 × 2 of prediction in the training data

```
svm.train.rad$fitted
table(svm.train.rad$fitted, train.y)
```

We employed cross-validation and determined SVMs with lowest 10-fold cross-validated overall error fraction in the training data. As the tuning function currently does not allow the output of other than the overall error rates, sensitivity and specificity were not estimated from cross-validation. Sensitivity and specificity were estimated in the training data. For the use of a sigmoid, radial, and polynomial kernel function, the following parameters were selected:

sigmoid: $K(\boldsymbol{x}, \boldsymbol{x}_i) = \tanh(0.0055\langle \boldsymbol{x}, \boldsymbol{x}_i\rangle + 0.1),\ \text{cost} = 0.95$

radial: $K(\boldsymbol{x}, \boldsymbol{x}_i) = \exp\left(-\frac{1}{2 \cdot 125}\|\boldsymbol{x} - \boldsymbol{x}_i\|\right),\ \text{cost} = 1.05$

polynomial: $K(\boldsymbol{x}, \boldsymbol{x}_i) = (0 + 0.02\langle \boldsymbol{x}, \boldsymbol{x}_i\rangle)^1,\ \text{cost} = 1.1.$

The SVMs with these kernel functions were then tested in the temporal and external validation data, estimating overall accuracy fractions as well as sensitivity and specificity. The respective results are depicted in Table 3.

To predict outcome in the test data using the data matrix test.x with features in the test data, to show predicted values for the test data and the 2 × 2 table of predictions in the test data, the following commands are used:

```
svm.test.rad <− predict(svm.train.rad, test.x)
svm.test.rad
table(svm.test.rad$fitted, test.y)
```

5 Learning machines: ensemble methods

Ensemble methods, or so-called *committee methods* represent a family of procedures that differs from all of the above. The basic idea is motivated by common real life experience: it is often better to consult several experts instead of just one. If the experts have different opinions, information is often obtained with greater reliability, the argument being that the different experts each see different parts of the sample space with greater individual efficiency. Usually, people seeking advice from these committees of experts follow a majority vote over the separate decisions. This is the principle utilised in ensemble methods although refined versions of the majority vote can be even better than simple majority vote (e.g., Schapire, 1990). Moreover, the predictions of a committee of quite distinct algorithms is provably at least as good as the best single algorithm as shown by Mojirsheibani (1999).

As in real-life committees, two conditions need to be fulfilled for the resulting classification to be more accurate than the single classifiers. Firstly, there has to be a certain amount of disagreement or lack of correlation among the committee members. Thus, if one classifier makes a wrong decision, it is possible that he or she will be outvoted by the majority. The underlying idea is that we would like the separate classifiers to efficiently see or interpret different aspects of the data. Secondly, the individual members have to be minimally accurate in their classifications, i.e., their error fraction must be better than random coin tossing. Otherwise, the error fraction of the majority vote will increase compared to the single error fractions.

An example makes this clear. Consider an ensemble consisting of 20 *independent* learning machines with an error fraction of 0.3 each. If a majority of more than 10 members is required for classification, then the error fraction can be calculated under a binomial distribution to be 0.017 instead of the individual 0.3.

In principle, all manner of single algorithms can be combined, including CART, SVMs, or classical logistic regression. However, simple algorithms are employed in practice, and in most instances, classification trees are employed. Indeed, considering ensembles of SVM or multivariable logistic regression as base classifier would be rather unusual.

However, the main question is how a committee of different experts can be created from the same single data set. Because the data are comprised of

- a number of cases
- a number of variables

different data sets can be created from it by manipulating either the cases, or by manipulating features, e.g., by randomly drawing predictors. For the first, some approaches, such as those that invoke boosting, modify the samples for later classifications by the errors of the earlier ones. The actual classification is then based on some combination of the different single classifications. In other procedures like bagging, which includes random forests as special case, bootstrap samples are drawn for every single classification. As an alternative, one can increase variability between experts by manipulating the input features or the outcome, or by introducing randomness in the learning algorithm (Dietterich, 2000).

Here, we focus on the ensemble methods of boosting (Section 5.1), bagging (5.2), and random forests (5.3). In addition to other ensemble methods including *Bayesian averaging* (Domingos and Pazzani, 1997; Hoeting et al., 1999; Hoeting, 2002), different further developments of the described algorithms like *bundling* (Hothorn and Lausen, 2005), *double-bagging* (Hothorn and Lausen, 2003), or *bagboosting* (Dettling, 2004) have been proposed. As the general principles of these resemble those described below, we refrained from describing them in detail.

5.1 Boosting

Boosting is a broadly useful ensemble method that sequentially applies a number of learning machines to repeatedly modified versions of the training data. An idea closely related to boosting traces back to Tukey (1977). He proposed ‘twicing’, where in a first step a linear regression model is fit to the original data. In a second step, the same linear model with the same predictors is fit to the residuals obtained after the first step. The parallel here with boosting is that in a linear model the residuals form a weighted version of the original data. With this two-step regression procedure, model fit and prediction can be improved if there is some recoverable structure left in the data after the first regression step. Furthermore, nothing is lost if residuals have zero mean after the first step. In the last 15 years, this line has been followed in a number of approaches.

The boosting procedure

For a sample of n observations, the original boosting idea as proposed by Schapire (1990) is as follows:

- Using the original n observations, develop a classification rule G_1 from a single learning machine, e.g., CART, SVM or logistic regression to predict the binary outcome $\{-1; +1\}$. Although most individuals will be correctly classified by this learning machine, several subjects will be misclassified, and it is possible that these patients systematically differ from those being correctly classified.
- Draw a random sample of total size n with replacement consisting of $\frac{n}{2}$ misclassified individuals and an identical number of subjects from the set of correctly classified patients. Train on the same features as before to predict the same endpoint; the result is the classifier G_2.
- Interesting cases are those with discrepant results of the first and the second learning machine. Therefore, train classifier G_3 with n cases drawn with replacement only from those for which G_1 and G_2 disagree.
- The boosted classifier is the majority vote across the three prediction rules G_1, G_2, and G_3.

This original boosting algorithm obviously does not use all available data in the second classification step and has therefore been improved over the last decade. In further developments, a sequence of M classifiers $G_1(x), \dots, G_M(x)$ is produced instead of three, and again, the predictions are then combined via a weighted majority vote to produce the final prediction. More generally, cases who were incorrectly predicted by previous classifiers are assigned a higher weight for the next iteration.

In very general terms, with M machines $G_1, G_2, \dots, G_M$, the ensemble is defined by a linear combination of the single machines $G_m(x)$:

$$G(\mathbf{x}) = \sum_{m=1}^{M} \alpha_m G_m(\mathbf{x})$$

with α_m being the coefficient associated by ensemble member G_m, and where α_m and G_m are evaluated within the procedure (Meir and Rätsch, 2003). The global aim of the learning procedure is to choose the machines G_m and weights α_m in such a way that a specific loss function is minimised. Two algorithms in which G_m and α_m are estimated are described below.

The most prominent boosting approach is the *Ada*ptive *Boost*ing procedure (Freund and Schapire, 1997), AdaBoost for short. It has already been utilised in different areas of medical research (e.g., Huang and Murphy, 2004; McLaughlin and Berman, 2003; Qu et al., 2002). The *discrete AdaBoost* algorithm works as follows (Hastie et al., 2003):

1 Initialise observation weights $\omega_{i,1} = \frac{1}{n}$ for $i = 1, 2, \dots, n$.

2 For $m = 1$ to M do

  a Fit a classifier $G_m(\mathbf{x})$ to the training data using the above weights.

  b Compute

  $$err_m = \frac{\sum_{i=1}^{n} \omega_{i,m} I(y_i \neq G_m(\mathbf{x}))}{\sum_{i=1}^{n} \omega_{i,m}},$$

  where $I(\cdot)$ denotes the indicator function.

  c Compute

  $$\alpha_m = \log \frac{1 - err_m}{err_m}.$$

  d For $i = 1, 2, \dots, n$ adjust weights

  $$\omega_{i,m+1} = \omega_{i,m} \exp\big(\alpha_m I\big(y_i \neq G_m(\mathbf{x})\big)\big).$$

3 Output $G(\mathbf{x}) = \operatorname{sign}\big(\sum_{m=1}^{M} \alpha_m G_m(\mathbf{x})\big)$.

Here, α_m is computed by the algorithm and fulfills a two-fold function. First, it determines the relative contribution of the respective G_m in the final step of combining all machines for an overall prediction. The idea is to assign greater weights to more accurate classifiers. And second, α_m contributes to data modifications by determining the weights $\omega_{i,m}$. As a consequence, cases that are misclassified receive more weight, and cases that continue to be misclassified receive increasing influence in the next iteration. The discrete AdaBoost either returns a -1 or a $+1$ as decision outcome, as seen from step 3 of the algorithm.

The main difference to further alternative procedures lies in the choice of the loss function. To be specific, AdaBoost uses an exponential loss function where

$$L(G) = E[\exp(-yG(\mathbf{x};\gamma)) \mid \mathbf{x}]$$

is minimised. Here, γ denotes the so-called regularisation parameter for the machine G (Hastie et al., 2003). In the variant called *LogitBoost* (Dettling and Bühlmann, 2003), one instead minimises

$$L(G) = E\left[y^* G(\mathbf{x};\gamma) - \log\left(1 + \mathrm{e}^{G(\mathbf{x};\gamma)}\right) \mid \mathbf{x}\right]$$

with $y^* = \frac{1}{2}(1+y) \in \{0;1\}$. Consequently, LogitBoost aims at fitting additive logistic regression models by a stepwise optimisation of the Bernoulli log-likelihood. If the outcome is now coded by $y^* \in \{0;1\}$ and the probability of the outcome $y^* = 1$ is given by $p(x)$ with $p(x) = \left(1 + \exp[-2G(\mathbf{x})]\right)^{-1}$, the algorithm of LogitBoost is given as follows:

```
1  Initialise observation weights $\omega_{i,m} = \frac{1}{n}$ for $i = 1, 2, \ldots, n$, $G_1(\mathbf{x}) = 0$ and
   $p(x_i) = \frac{1}{2}$.
2  For $m = 1$ to $M$ do
     a  Compute working response and weights
          $z_i = \frac{y_i^* - p(x_i)}{p(x_i)(1 - p(x_i))}, \quad \omega_{i,m} = p(x_i)(1 - p(x_i)).$
     b  Fit $g_m(\mathbf{x})$ by weighted least-squares regression of $z_i$ to $x_i$ using weights $\omega_{i,m}$.
     c  Update $G_{m+1}(\mathbf{x}) \leftarrow G_m(\mathbf{x}) + \frac{1}{2} g_m(\mathbf{x})$ and $p(x)$ via above formula.
3  Output sign $(G(\mathbf{x})) = \mathrm{sign}\left(\sum_{m=1}^{M} g_m(\mathbf{x})\right)$.
```

The choice of a different loss function, e.g., AdaBoost or LogitBoost, has been seen to depend on the noise in the data. Thus, with noisier data LogitBoost fares better, as AdaBoost tends to focus too much on outliers that are difficult to classify or on possibly unrepresentative data points. Other variants of the boosting scheme, including *Gentle AdaBoost* (Friedman et al., 2000) and *BrownBoost* (Freund, 2001), directly de-emphasise outliers when these turn out to be too difficult to classify.

Once the loss function and hence the specific form of the boosting algorithm has been chosen, one has to decide upon which base classifier to use and how many machines to combine. In practice, CART with small trees is usually employed as base learning machines G_m. This can be justified by the experience that boosting profits most from machines that already perform quite well but are slightly too simple for the data at hand. Hence, even stumps seem to work well in a boosting ensemble.

With regard to the number of machines to be combined, recent research has shown that there exists an optimal Bayes-consistent minimum number of machines to be trained (see Koltchinskii and Yu, 2004 and the associated papers). Of particular importance in this research is the fact that training more machines than this optimal number may steadily increase the error fraction. Boosting therefore may result in overfitting if an excessive number of machines is used, thus boosting can create ensembles that are less accurate than a single classifier (Opitz and Maclin, 1999). There has been discussion in the literature on whether boosting algorithms routinely overfit the data. Experience has shown that with smaller sample sizes and higher noise in the data, the procedure does indeed overfit. Therefore, some regularisation is required to limit the complexity of the model.

Similarities between boosting algorithms and SVMs have been shown lately (for an overview, see http://www.boosting.org). Both approaches attempt to somehow maximise the margin; however, the norms utilised for the instance vector and the weight vector differ. In addition, different optimisation procedures are used, and as a result, different margins are obtained.

Example for boosting

As an example of applying boosting algorithms, we utilised *LogitBoost* for R as implemented by Dettling which can be retrieved via http://stat.ethz.ch/~dettling. Here, stumps are grown as base learning machines. For internal validation, we used 10-fold cross validation in the training sample.

Logistic boosting using the package can be carried out with

```
boost.train <- logitboost(train.x, train.y, train.x, mfinal = 100)
```

where `train.x` and `train.y` denote the features and the outcome vector in the training data, and `mfinal` is the number of trees to be grown. Results from 100 stumps in training and internal validation data are displayed in Table 3. When varying the number of stumps from 20 to 100, only a negligible impact on the cross-validated error fraction was observed. If 100 trees are grown, the object `boost.train` contains 100 elements. Thus, the predicted probabilities after the last tree for the training data are stored in `train.probs` via

```
train.probs <- boost.train$probs[, 100]
```

We use a loop to classify patients according to the 50% probability cut-off and to create a 2 × 2 classification table in the training data

```
train.probs.bin <-0
for (i in 1 : length(train.probs))
    {train.probs.bin[i] <- if(train.probs[i] > 0.5) 1 else 0}
table(train.probs.bin, train.y)
```

10-fold cross-validation may be performed by boosting

```
boost.train.cv <- crossval(train.x, train.y, v = 10, mfinal = 100)
```

with v denoting the number of data splits. The predicted probability values for the cross-validated training data are obtained by

```
train.probs.cv <- boost.train.cv$probs[, 100]
```

As described above for the object `train.probs.bin`, patients may be classified with working object `train.probs.cv` instead of `train.probs`. Results from the training data can be tested on the test data `test.x` via

```
boost.test <- logitboost(train.x, train.y, test.x, mfinal = 100)
```

Predicted probabilities, classifications and the 2 × 2 table of classifications in the test data can be obtained with similar code as above. In fact, we applied the resulting model to the temporal and external validation data, and the respective error fractions are given in Table 3.

5.2 Bagging

Bagging was named after its property to aggregate the information from different bootstrap samples, and is therefore bootstrap aggregation or bagging for short. Similar to boosting, the general purpose is to sequentially apply a number of learning machines to repeatedly modified versions of the training data. Both bagging and boosting utilise a combination of the many individual classifiers to generate an overall prediction. However, there are two important differences. First, whereas in boosting different weights are assigned to different machines, each single machine typically receives the same weight in the ensemble classification in bagging. The second difference lies in the composition of the training samples for every machine. In boosting, training samples are formed in a sophisticated way. In contrast, in bagging, a random sample, typically a bootstrap sample, is trained for every machine of the original data set.

As each bagging step generates a disjoint pair of data sets, training is done on one of the pairs and testing on the other.

The bagging procedure

With CART being the base classifier, the 'standard' bagging procedure is:

- From the original training data T with n cases, a bootstrap sample T^* consisting of n cases is drawn with replacement. Hence, a specific case might be included once, several times, or not at all.
- T is separated into the bagged sample T^* and the out of the bag (OOB) sample $T\backslash T^*$ which are independent.
- A classification tree is grown on the bagged sample T^*.
- All patients who are part of the OOB sample $T\backslash T^*$ are dropped down the tree developed on the bagged sample for classification.
- The steps described above are repeated to train a pre-specified number of machines. For classification, simple majority voting is used.

The number of bootstrap samples to be drawn need not be fixed in advance and several may be tried. However, numerous examples in the literature have shown that most of the improvement in accuracy occurs after just 10 to 15 machines (Opitz and Maclin, 1999). We comment that with a sample size of n cases in the training set, the probability for a case just not to be drawn is $(1 - \frac{1}{n})$ in a single draw, and $(1 - \frac{1}{n})^n$ after n draws which is approximately $\mathrm{e}^{-1} = 0.328 = 1 - 0.632$ as n tends to infinity. Hence, each bootstrap sample contains, on average, about two thirds of the original training set. Immediately we see that one drawback of bagging, as with most bootstrap methods, is that each model, i.e., each individual classifier is generated using only about $\frac{2}{3}$ of the data: a reduction in effective sample size is always operating. As every ensemble machine is tested on an independent sample of OOB cases, there is, in principle, no need to further internally validate the final model in order to yield an unbiased estimate of the error fraction, estimated from the OOB samples.

An alternative to standard bagging, where bootstrap samples are drawn, is *SuBagging*. Here random samples, e.g., of half of the size of the original training data set are drawn without replacement (Bühlmann, 2003).

As bagging tends to mainly reduce the variance without greatly increasing the bias, it has been demonstrated that bagging improves the accuracy, especially if the combined classifiers are *unstable* (Breiman, 1996), i.e., if small changes in the data result in large differences in the classification. Phrased differently, bagging is most effective when the classifiers have a large variance. This applies, e.g., to CART, but also to regression models with subset selection. Consequently, for these approaches bagging is almost always more accurate than a single classifier. However, bagging often does not improve upon simple linear discriminant classifiers since these are known to be quite stable.

Example for bagging

For analysing our stroke data sets, we used the implementation by Peters and Hothorn as part of the package *ipred* in R (R Development Core Team, 2005). In this package, the single machines are CART. Using this approach we first grew maximally sized trees and varied the number of trees between 10 and 100. Specifically, the model is developed on the training data with `nbagg` bootstrap samples and the computation of the OOB error estimates `coob = TRUE` by

```
bag.model <–bagging(y ~ ., data = train, nbagg = 50, coob = TRUE)
```

Here, `train` is a data frame containing the features and a vector `y` of outcome factorised in $\{-1; +1\}$ in the training data.

We next determined the accuracy in the training and the test data set (see Figure 9). Here, the accuracy did not increase visibly after drawing about 25 bootstrap samples. The minimal observed overall error fraction in the training sample of 0.2159 was obtained with 50 trees, and the respective overall error fraction, sensitivity, and specificity are shown in Table 3. As the resulting model is already internally validated via the incorporated bootstrapping procedure, no separate cross-validation was performed. For temporal and external validation, the tree ensemble was applied to classify patients in the test sample. Again, results are displayed in Table 3 and Figure 9.

```
pred.train <–predict(bag.model, type = "class",
                     aggregation = "majority")
```

gives the predicted values for the training data, where `type = "class"` indicates that classification is returned and `aggregation = "majority"` specifies the aggregation algorithm. Finally,

```
pred.test <-predict(bag.model, newdata = test.x, type = "class",
                        aggregation = "majority")
table(pred.test, test.y[, 1])
```

yield the predicted values for the test data and the corresponding 2 × 2 table.

5.3 Random forests

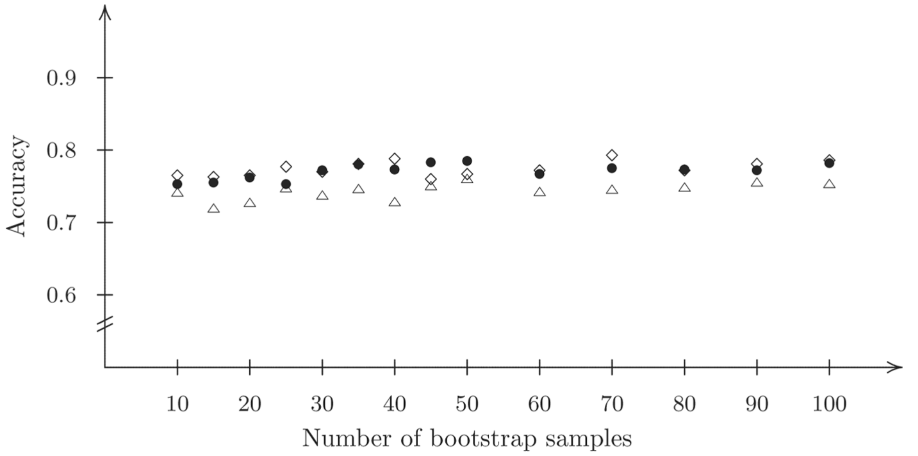
A modification of bagging is random forests which has been introduced by Breiman (2001). Applications of random forests in medical research have mostly focused on the classification of genetic data (e.g., Schwarz et al., 2007; Schwender et al., 2004; for an overview, see Ziegler et al. (2007)). As the name implies, the basic units of this method are trees, and it utilises a combination of manipulating the training cases together with introducing an additional element of randomness. To be specific, a number of trees are grown on bootstrap samples of all available cases as in bagging. However, only a random selection of available features is used for each node within each tree so that an element of random searching in the feature space is introduced. A subject is classified according to the majority vote across all trees. Thus, random forests are a combination of tree classifications where each tree depends on the values of a random vector sampled independently and with the same distribution for all trees in the forest (Breiman, 2001).

The random forests procedure

In detail, the standard random forest procedure takes the following steps:

- A bootstrap sample T^* consisting of n subjects is drawn from the original training data T with n cases.
- For this sample T^*, a CART is grown. However, unlike in CART, not all R available features are used. Instead, at each node, a small number r of features is randomly selected without replacement, and the split is based on the best of the r variables: this is termed *random feature selection*. Although at each node, different variables might be selected to be tested, the number r is held constant during the procedure, and the default usually is the square root of the available number of features.
- Unlike in CART, no pruning is used, and the tree is grown to its largest possible extent. Another difference is that not every case is dropped down the tree, but only every patient who belongs to the OOB sample $T \backslash T^*$. Hence, for growing and classification, independent samples are used in the generation of a single tree.
- These are repeated to grow a pre-specified number of trees. For classification, the majority vote over all trees in the resulting forest is used.

Figure 9 Accuracy of the bagging procedure in the training data (solid bullets), temporal (open diamonds) and external (open triangles) validation data. As basic algorithms maximal-sized classification trees were grown. The number of bootstrap samples varied from 10 to 100



The error fraction of the resulting forest has been shown to depend mainly on the correlation between any two trees in the forest: Increasing the correlation also increases the error fraction. Naturally, the error fraction hinges upon the error fractions of every individual tree in the forest. Reducing the number r of variables selected at each node reduces both the correlation and the strength of accuracy. Hence, although the default setting for r is the square root of the number of available predictors, varying this number is encouraged, and the range of optimal values is usually quite wide.

The second parameter that needs to be set in advance to grow a random forest is the number of trees included. No clear recommendations have been given; however, the more trees are grown, the more stable the results will be, and the upper limit seems to be merely a computational issue, as it has been observed that random forests do not generally overfit as more trees are added (Breiman, 2001). Furthermore and in contrast to boosting, Bayes consistency has recently been demonstrated for a simple version of random forests (Breiman, 2004).

A special feature of random forests is that the individual importances of the predictive variables in the final model can be estimated. Two importance measures have received attention because they are implemented in standard software for random forests; a detailed discussion of these and other importance measures has been given by us (Ziegler et al., 2007). One is the *permutation importance*, where the first step is to calculate the number of correct classifications of the OOB subjects in every single tree grown in the forest. Secondly, the values of the jth feature are randomly permuted in the OOB subjects, and these subjects are then re-classified using these new values. Finally, the number of correct classifications with the permuted values is compared with the number of correct classifications in the original data. The difference between these fractions, averaged over all trees in the random forest, gives the permutation importance for the jth variable. The second importance measure is a generalisation of the Gini index from a single tree to a forest. The basic idea of this importance measure is to contrast the impurity of a tree with and without the feature of interest being included in the tree; for details, see Ziegler et al. (2007). If the estimated importance of all features can be assumed to be independent from tree to tree, a standard error of the importance can be computed in a usual way so that asymptotic confidence intervals assuming normality can be calculated (Lin et al., 2004).

Importance measures are useful for feature selection (Ziegler et al., 2007). If the number of variables is large, forests can be run once with all features, then run again using only the most important ones from the first run (Diaz-Uriarte and Alvarez de Andres, 2006). However, the importance measures may be biased when different features have different scales or different numbers of categories. This bias can be avoided by using subsampling without replacement instead of bootstrapping (Strobl et al., 2007).

Another option available with random forests for interrogating the data is the calculation of proximities between subjects. For this, after a tree is grown, every subject is dropped down each tree. Then, each pair of subjects is compared with regard to their final stopping point. That is, if they are classified into the same final node in a single tree of the forest, the proximity between them is increased by one. The resulting values could, in principle, be used to replace missing data and to identify outliers.

As in bagging, there is no need for cross-validating random forests to get a more unbiased estimate of the generalisation error, as every included tree is tested in the independent OOB subjects.

Example for random forests

For our stroke example data, we utilised the *R* implementation *RandomForest* by Liaw and Wiener (2007). The syntax is similar to the syntax described above. To be specific, random forests are trained by:

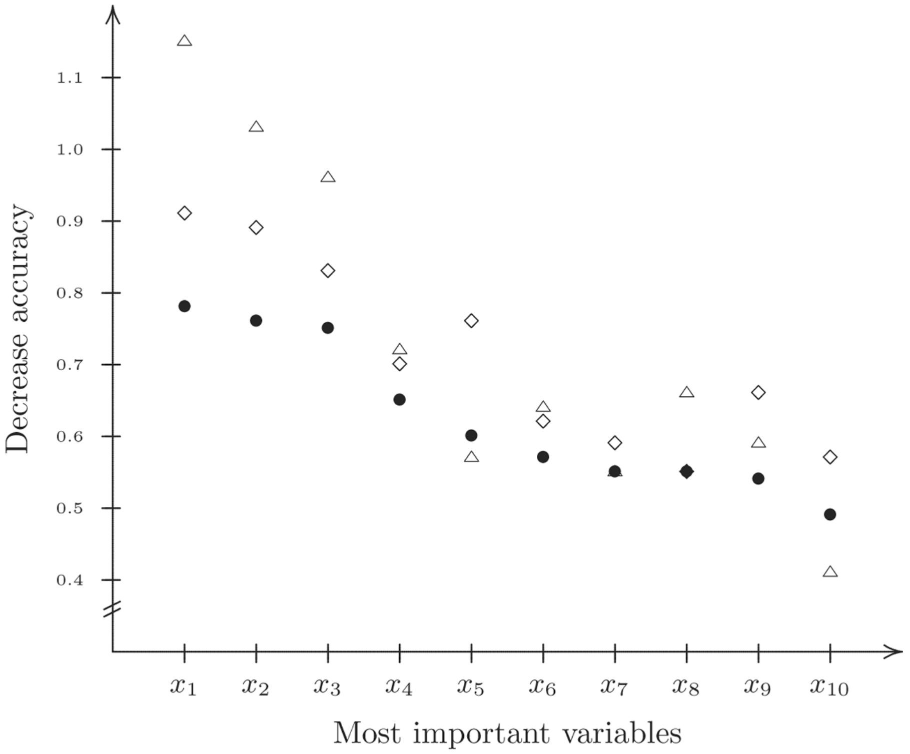
```
rf.train <−randomForest(train.x, train.y, ntree = 500, mtry = 11,
                        importance = TRUE)
```

As before, `train.x` and the vector `train.y` denote the training data. `ntree` is the number of trees to be grown, `mtry` gives the number of features in every split. Finally, `importance = TRUE` sets the output of the Gini importance which may be displayed via `rf.train$importance`. To apply the estimated random forests to the test data, both the features and the outcome for the test data are utilised:

```
rf.test <−randomForest(train.x, train.y, xtest = test.x,
                       ytest = test.y,
                       ntree = 500, mtry = 11, importance = TRUE)
```

We varied the number of variables to be selected at each node in a single tree between 2 and 11. As the error fractions in the training data were not greatly affected by this, the forest resulting from the default value of $\sqrt{38} \approx 7$ is presented. As an additional option, equal weight was given to the two patient groups. The number of trees to be grown was set to 500, and the importance of each variable was calculated. Figure 10 shows the importance of the ten most important predictors. In detail, the importance was calculated for the entire data set as well as separately within patients who were completely restituted or incompletely restituted. Interestingly, the ten important variables depicted in Figure 10 for random forests are very similar to the variables identified with logistic regression. The resulting error fractions in the training data as well as from applying the developed forest to temporal and external validation data are given in Table 3.

Figure 10 Importance of the ten most important variables in the random forest grown in the stroke training data. x_1 = Age, x_2 = Rankin scale, x_3 = NIH-SS total score, x_4 = Right arm paresis, x_5 = Fever, x_6 = Left arm paresis, x_7 = Left leg paresis, x_8 = Right leg paresis, x_9 = Neurological complications, x_{10} = NIH-SS questions. A random selection of $\sqrt{38} \approx 7$ variables was used at each node, and a number of 500 trees were grown in total. Solid bullets represent mean importance for overall classification, open diamonds and triangles stand for importance for predicting functional independence and dependence or mortality, respectively



6 Comparison of learning machines

In the previous sections we have described the theoretical background of different learning machines as well as their application to clinical prognostic data. However, so far we have postponed a direct comparison of the machines against each other. How this aspect of prognostic modelling building is undertaken differs sharply in the clinical context from how it is usually handled in the machine learning community, for four reasons. First, with clinical data we do not usually have a sufficient understanding of the distributional structure of the data to enable theory-based performance comparisons or useful generation of synthetic data. Second, the sample sizes, especially in the case, or at-risk group is often rather small. Hence we must more carefully consider sample variances of the error fraction estimates. Third, it is important to account for the correlation between error fraction estimates, as these are built using paired binomial outcomes. Finally, fourth, for accurate machine comparisons with medical data we find that different criteria should be considered, criteria distinct from those usually considered in the machine learning setting:

1 Accuracy of the model in temporal and external validation data
2 Ease of application of the algorithm and computational burden
3 Clinical interpretability of the resulting model.

These criteria will be discussed in the following sections.

6.1 Accuracy of predictions

The most important questions are: Do the error fractions between two machines differ in a relevant magnitude and/or significantly from one another? As explained by Hand (2006), one cannot expect a pronounced superiority of a highly sophisticated method. Simple methods are often equally effective or even superior in classifying new data points.

For statistical analysis, one has to note that in practice different machines are built on data from the same patients (Newcombe, 1998). In addition, as mentioned above, the predictions from two machines for the same patient can be expected to be correlated, i.e., outcomes for paired machines fit the matched pair design. Therefore, in classification problems, a significance test for differences in error fractions, sensitivity or specificity between two machines A and B can be based on McNemar's test. To be specific, with the notation from Table 2 the null hypothesis H_0: $\delta = q_{12} - q_{21} = 0$ is tested against the alternative H_1: $\delta \neq 0$.

Table 2 Comparison of observed error frequencies (theoretical fractions in parenthesis) for two dependent machines

		Machine B					
		Correct		*False*		*Total*	
Machine A	Correct	a	(q_{11})	b	(q_{12})	$a+b$	(π_A)
	False	c	(q_{21})	d	(q_{22})	$c+d$	$(1-\pi_A)$
	Total	$a+c$	(π_B)	$b+d$	$(1-\pi_B)$	n	(1)

Tango (1999) has derived a simple asymptotic $(1-\alpha)$-confidence interval for the difference δ that has good power and coverage probabilities. It is given by

$$\frac{b-c-n\delta}{\sqrt{n(2\hat{q}_{21}+\delta(1-\delta))}} = \pm z_{\frac{\alpha}{2}}$$

with $z_{\frac{\alpha}{2}}$ denoting the upper $\frac{\alpha}{2}$-quantile of the normal distribution. Here, $\hat{q}_{21}$ is given by the maximum likelihood estimator for q_{21}:

$$\hat{q}_{21} = \frac{\sqrt{(-b-c+(2n-b+c)\delta)^2 - 8n(-c\delta(1-\delta))} - \left[-b-c+(2n-b+c)\delta\right]}{4n}.$$

Alternatives include the recently proposed confidence interval by Zhou and Qin (2005, 2007); for an overview see Newcombe (1998). In our example, we employed Tango's method for computing confidence intervals for correlated outcomes.

For a comparison of error fractions between independent data sets, for example between data sets for training, temporal or external validation, tests and confidence intervals with improved power and coverage have been developed recently (Newcombe, 1999; Brown and Li, 2005). Specifically, we employed the improved confidence interval described as method 10 (Newcombe, 1999) for comparing independent samples.

One aspect in comparing classifications from different machines deserves specific attention. If various machines are trained on training data, their prediction accuracy can be compared in a fair manner by applying all machines to the validation data. In this case, the above described procedures lead to valid estimates. If, however, machines are trained and compared on the same single data set, prediction accuracy may vary systematically depending on the way machines are trained. For example, the logistic regression and support vector machines utilise all available data in the model building step and are more prone to overfitting if appropriate adjustments are not carried out. Thus, error fractions may be underestimated.

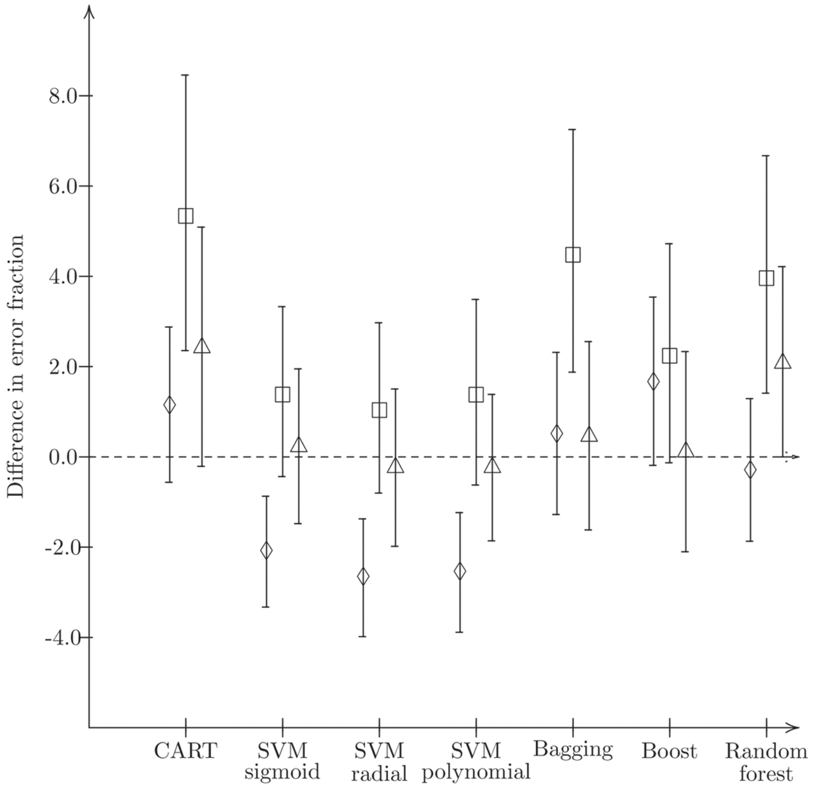
In comparison, ensemble methods use the available data only partly and rarely overfit. Therefore, error fractions are more reliable, and thus, unfortunately, often higher. To be specific, the prediction of a single patient in algorithms that utilise bootstrap samples is based on only about two thirds of the entire training data set. By contrast, even when a 10-fold cross validation is employed for the other algorithms, 90% of the training data is used in the prediction of a single patient. To overcome this source of bias in comparing prediction accuracies, in principle, bootstrap samples could be drawn in the first step. In the second step, all machines to be compared could be trained on the bootstrap samples and tested on the out of bag samples. This gives paired classification results and bootstrap sample confidence intervals for all machines. These could then be compared by appropriate averaging across all bootstrap samples. In our own analyses, we did not follow this approach for the training data due to CPU – time restrictions.

For a statistical comparison of any pair of two machines, exact McNemar's tests were calculated. In addition, 95%-confidence intervals for the difference in error fractions according to Tango (1998) were determined utilising an *R* implementation (http://www.stat.ufl.edu/~aa/cda/R/matched/R2_matched). As an example, the difference in error fractions and the respective confidence intervals comparing every learning machine with the logistic regression are depicted in Figure 11. It can be seen that the accuracy of CART, bagging, and random forests is significantly worse than the logistic regression in the temporal validation data. In contrast, logistic regression is worse than the SVMs in the cross-validated training data, but no difference is visible with regard to temporally or externally validated error fractions. No significant differences are detected between logistic regression and boosting although error fractions are higher for boosting in the cross-validated and temporally validated data sets.

Example for comparison of learning machines

Table 3 shows a summary of the accuracy fractions of all the learning machines described above including the original logistic regression. To give a more precise picture, fractions within all patients as well as within completely restituted (specificity) and incompletely restituted or deceased (sensitivity) cases are presented. For all machines, fractions in the training data set, in the temporal and the external validation data set are shown as well. In addition, internally validated fractions are given for logistic regression, CART, and boosting.

Figure 11 Difference in error fractions in internally validated training (diamonds), temporal (squares), and external (triangles) validation data compared with logistic regression. Mean differences in error fractions and 95% confidence intervals according to Tango (1998) are presented



As expected, for each learning machine, fractions of overall accuracy drop when moving from training fractions via internal and temporal to external validation fractions. However, each machine was able to correctly classify more than 70% of the patients even in the external validation sample; logistic regression, SVM, bagging and boosting even exceeded an accuracy fraction of 75%. A further consistent trend is that the externally validated specificity was always much higher than the respective sensitivity, indicating that fully recovered patients were easier to classify.

Comparing the machines with each other, it seems that whereas in the training data, SVMs fared best, the picture is different in both validation data sets: Here, logistic regression and SVM yield the highest accuracy fractions. The complete comparison of the error fractions between the machines together with 95%-confidence intervals for the differences are given in Table 4 for the training and in Table 5 for the validation data.

Table 3 Resulting accuracy fractions in the stroke data

		n	*Logistic regression*	*CART*	*SVM sigmoid*	*SVM radial*	*SVM polynomial*	*Bagging*	*Boosting*	*Random forest*
Training data	Overall	1737	80.89	78.93	81.00	81.58	81.46	78.41	78.58	79.22
	Sens	722	77.01	70.78	68.01	70.08	71.05	72.58	73.55	70.91
	Spec	1015	83.65	84.73	90.25	89.75	88.87	82.56	82.17	85.12
Internal val	Overall	1737	79.09	77.75	n.c.	n.c.	n.c.	n.c.	77.26	n.c.
	Sens	722	61.46	68.83	n.c.	n.c.	n.c.	n.c.	72.85	n.c.
	Spec	1015	91.36	83.98	n.c.	n.c.	n.c	n.c.	80.39	n.c.
Temporal val	Overall	581	81.07	75.73	79.69	80.03	79.69	76.59	78.83	77.11
	Sens	250	70.40	49.60	69.20	70.00	72.80	59.60	62.80	58.40
	Spec	331	89.12	95.47	87.61	87.61	84.89	89.43	90.94	91.24
External val	Overall	866	76.33	73.90	76.10	76.56	75.40	75.87	76.21	74.25
	Sens	376	67.55	49.47	70.21	70.74	72.61	63.03	62.23	60.90
	Spec	490	83.06	92.65	80.61	81.02	77.55	85.71	86.94	84.49

Sens: Sensitivity, Spec: Specificity, val: Validation, n.c.: not calculated.

Table 4 Difference in overall accuracy fractions in the stroke data. Shown are the differences in accuracy of machine on the left minus on the top in the training with respective 95% confidence intervals

	CART	*SVM radial*	*SVM sigmoid*	*SVM polynomial*	*Bagging*	*Boosting*	*Random forest*
Logistic regression	1.96 (0.21; 3.67)	−0.69 (−2.07; 0.67)	−0.12 (−1.51; 1.26)	−0.58 (−1.84; 0.67)	2.48 (0.74; 4.18)	2.30 (0.59; 3.99)	1.67 (0.14; 3.17)
CART		−2.65 (−4.32; −1.00)	−2.07 (−3.78; −0.39)	−2.53 (−4.25; −0.85)	0.52 (−1.17; 2.17)	0.35 (−1.05; 1.72)	−0.29 (−1.78; 1.19)
SVM radial			0.58 (−0.06; 1.21)	0.12 (−0.58; 0.81)	3.17 (1.51; 4.79)	2.99 (1.29; 4.67)	2.36 (0.97; 3.73)
SVM sigmoid				−0.46 (−1.23; 0.30)	2.59 (0.87; 4.28)	2.42 (0.71; 4.10)	1.78 (0.38; 3.17)
SVM polynomial					3.05 (1.36; 4.71)	2.88 (1.16; 4.56)	2.25 (0.82; 3.64)
Bagging						−0.17 (−1.95; 1.58)	−0.81 (−2.26; 0.63)
Boosting							−0.63 (−2.20; 0.91)

Table 5 Difference in overall accuracy fractions in the stroke data. Shown are differences in overall accuracy in the temporal (upper triangle, difference is accuracy of machine on the left minus on the top) and external validation data (lower triangle, difference is accuracy of machine on the top minus machine on the left) with respective 95% confidence intervals

	Logistic regression	*CART*	*SVM radial*	*SVM sigmoid*	*SVM polynomial*	*Bagging*	*Boosting*	*Random forest*
Logistic regression		4.33 (1,71; 6.88)	1.91 (−0.06; 3.83)	2.95 (0.81; 5.03)	1.91 (0.13; 3.65)	3.12 (0.65; 5.52)	1.73 (−0.44; 3.85)	2.25 (−0.05; 4.50)
CART	2.55 (0.28; 4.76)		−2.43 (−4.90; −0.01)	−1.39 (−3.71; 0.89)	−2.43 (−4.80; −0.11)	−1.21 (−3.85; 1.36)	−2.60 (−4−89; −0.35)	−2.60 (−4.99; −0.26)
SVM radial	1.27 (−0.45; 2.97)	−1.27 (−3.58; 0.98)		1.04 (−0.64; 2.69)	0.00 (−1.61; 1.58)	1.21 (−1.14; 3.51)	−0.17 (−2.37; 1.98)	−0.17 (−2.02; 1.64)
SVM sigmoid	0.69 (−1.11; 2.46)	−1.85 (−3.95; 0.20)	−0.58 (−2.08; 0.90)		−1.04 (−2.40; 0.31)	0.17 (−2.13; 2.43)	−1.21 (−3.46; 0.99)	−1.21 (−3.18; 0.72
SVM polynomial	0.35 (−1.29; 1.95)	−2.20 (−4.38; −0.06)	−0.93 (−2.33; 0.46)	−0.35 (−1.66; 0.95)		1.21 (−1.09; 3.46)	−0.17 (−2.32; 1.93)	−0.17 (−2.08; 1.70)
Bagging	2.66 (0.63; 4.65)	0.12 (−2.14; 2.33)	1.39 (−0.61; 3.34)	1.97 (−0.14; 4.03)	2.31 (0.22; 4.37)		−1.39 (−3.91; 1.08)	−1.39 (−3.26; 0.46)
Boosting	1.50 (−0.78; 3.73)	−1.04 (−2.70; 0.60)	0.23 (−1.97; 2.38)	0.81 (−1.35; 2.93)	1.16 (−0.97; 3.24)	−1.16 (−3.40; 1.04)		0.00 (−2.17; 2.13)
Random forest	1.97 (−0.01; 3.91)	−0.46 (−2.61; 1.64)	0.81 (−0.92; 2.51)	1.39 (−0.35; 3.10)	−0.02 (−0.04; 0.00)	−0.58 (−2.27; 1.09)	0.58 (−1.48; 2.60)	

6.2 Ease of application and computational time

The ease of applying a specific algorithm depends mainly on the availability of implementations and on the number of parameters that need to be sensibly set. As described in the context of our example, all of the described learning machines have been implemented in the software package *R* (*R* Development Core Team, 2005) which is available under the GNU public license. Even though the logistic regression modelling in our example was performed using SAS 8.02, *R* implementations are available, too.

In contrast, the difficulty with regard to reasonably setting parameters in advance differs widely across the different machines. For growing classification trees with CART, only the decision on the splitting criterion and on whether or not to prune the tree needs to be made. Similarly easy is the employment of the ensemble methods described above: For bagging, only the base learning machines and the number of bootstrap samples require selection. Because of this ease of use and accuracy, (Breiman, 1996) stated that bagging "goes a way toward making a silk purse out of a sow's ear, especially if the sow's ear is twitchy". The same holds for boosting: once the base classifier is chosen, the action of the boosting algorithm is mainly determined by the cost function, e.g., by utilising AdaBoost vs. LogitBoost. In random forests, the free parameters are again the number of machines and, in addition, the number of variables to be selected at every node. Importantly, our example underscores the fact that these algorithms are not overly sensitive to variations in the parameter values, number of machines, or number of variables.

The only learning machine described which requires an intensive tuning effort are the SVMs. Here, applying different kernel functions can lead to drastically different accuracy fractions. In addition, each kernel function necessitates the optimal adjustment of several parameters. In a similar way, the logistic regression approach utilised before required several parameters, for instance, algorithm and criteria for selecting variables and handling of interactions (Harrell et al., 1996; Royston and Sauerbrei, 2004).

Concerning CPU time, most of the learning machines do not differ remarkably with current computational capacities. The only exceptions were the SVMs, for which higher computational requirements have also been described before (Freund and Schapire, 1999). In summary, logistic regression and SVM are rather difficult to employ since they require significant tuning; other learning machines do not require such effort at CPU–time, or deeper mathematical preliminaries.

6.3 Clinical interpretability

The importance of this criterion lies in the experience that a proposed predictive model needs to be clinically meaningful in order to be widely accepted in practice. More specifically, factors that may affect the clinical credibility are as follows (Wyatt and Altman, 1995):

- For inclusion in the model, all data that are clinically relevant have been tested.
- All the data that are included in the final model are easy to obtain reliably in clinical practice.
- For continuous variables, arbitrary thresholds are avoided.

- The structure of the model is transparent, and the resulting predictions make sense. Phrased differently, the model is not just a 'black box'.
- It is easy to calculate the model's prediction for a specific patient.

The first bullet is not specific to machine learning algorithms employed but refers to the design of the data collection in advance. The second includes both the kind and the number of variables being included in the final model. Whereas the type of variable should be already determined in the first step and does not depend on the modelling algorithm, the number of variables can differ widely. Furthermore, it often is desirable to select variables and thus generate a parsimonious prediction model. In CART, the selection of important variables is clearly visible in the resulting tree. However, variables selected higher in the tree are not necessarily more important overall. More complicated but still possible is a variable selection in random forests: By determining the importance of every included variable, it is feasible to select the most important variables and re-build a random forest using only these. The option of variable selection also is an inherent feature in logistic regression modelling. However, the importance of the variables can only be assessed by leaving them out – or plugging them in – during the modelling process. Approaches that utilise this technique have also been proposed for SVMs (see, e.g., Guyon and Elisseeff, 2003). However, there is a clear need for improvements. In contrast, for bagging and boosting, we are not aware of any published variable selection approaches though extensions for appropriate base learners seem to be quite obvious.

The handling of continuous variables, bullet 3, depends on the learning algorithm: While logistic regression and SVM make use of the entire quantitative information, CART explicitly does not. Thus, for all ensemble methods based on trees, continuous variables are split optimally with regard to the reduction of purity, e.g., deviance at a specific node. Hence, the threshold might be statistically sensible but may appear clinically arbitrary.

In our view, the most important factors for clinical credibility are bullets 4 and 5 relating to the transparency and intuition of the resulting model. For logistic regression, the prediction model is expressed in a closed formula including only the relevant variables and calculating the predicted outcome. Hence, it is clearly visible which variables contribute in what way to the prediction. It is therefore easy to calculate the predicted outcome for a specific patient. Similarly, a CART generated tree is relatively easy to interpret. The results can be displayed in a flow chart as in Figure 4. The inherent 'logic' in the tree is apparent, and a single patient can be classified very easily by dropping the patient's data vector down the tree. The situation is different for SVMs and ensemble methods.

Although the functioning of bagging, boosting, and random forest might be transparent, the resulting model most likely is not, as the classification can no longer be summarised in a simple and interpretable structure. As a consequence, the prediction model rather resembles a 'black box' where the "internal workings are not readily inspected or understood by a user or system developer" (Hart and Wyatt, 1990, p.229). Implications of this include that clinical explanations are limited and little insight is gained into which variables are important for the prediction.

One possible strategy to recover interpretability lies in the sensible combination of methods. For example, one might use random forest on a full set of features to identify the most important variables and then send this small list to a logistic regression model which then lends itself readily to interpretability (Schwarz et al., 2007). In detail, the following strategy seems to be reasonable (see, e.g., Ziegler et al., 2007):

- Random forests are grown on the data using all available predictive variables with an implementation of its variable importance feature.
- A smaller number of the five or ten most important variables are selected. Random forests are grown on this subsample of features.
- The prediction accuracy is compared using the confidence intervals for dependent samples (Newcombe, 1998; Tango, 1999; Zhou and Qin, 2005, 2007).

The strategy is, therefore, to acquire good prediction in the first sequence with all available features, and then to look for structure in reduced models. There are limits, of course, in that the relative size of a data set may not generate sufficient statistical power to distinguish any two models, and an enormous initial variable list may make nearly invisible any single useful feature. Furthermore, this stepwise selection process might result in bias and/or overfitting. On the other hand, the strategy of good prediction first can be simplified by using an ensemble approach to a collection of diverse engines: we need not confine our prediction approach to any single machine.

7 Discussion

Over the last several years, many new algorithms have been suggested by the machine learning community that offer interesting alternatives for the development of multivariate prognostic models. One important difference to the classical logistic regression is that the latter is based on regularity or distributional conditions that may not hold, or, more likely, will simply not be testable. In addition, the classical method may not lend itself easily to a systematic inclusion and analysis of higher order terms or interactions among the features. In contrast, the machines studied here are intensely nonparametric, meaning that no assumptions are made at the outset about the structure of the data, or the correlation structure among the features. Put simply, it can be said that these methods simply do not see the likelihood function at all.

In comparing accuracy across different machines we saw that logistic regression, bagging, boosting, and random forests fared quite similarly in the training data with the SVMs being significantly better. However, it has to be kept in mind that we utilised cross-validated error fractions for logistic regression and boosting, and that bagging and random forests incorporate an estimation of internally validated accuracy as part of the methodology. In contrast, the original estimated training rates were used for SVMs. Hence, the presumably higher accuracy of the SMVs might be explained by overfitting.

In the temporal validation data, the lowest error fractions were obtained using logistic regression. The worst accuracy were found for CART, bagging, and random forests. Interestingly, no significant differences were detected with regard to accuracy rates in the external validation data set. However, both CART and random forests yielded higher error fractions than the other machines, while the other algorithms seem to be comparable. In addition, a remarkable difference between sensitivity and specificity is obvious for most machines with the specificity being much higher in temporal and external validation data, this difference is less than 20% for logistic regression and SVMs.

To summarise, with regard to the accuracy of the model in the German Stroke data no learning machine among those studied here strictly outperformed our original logistic regression approach (König et al., 2007). The best results overall were obtained with the logistic regression and the SVMs.

Our experience has shown that the price to be paid for this high accuracy is the user-parameter tuning effort and computational burden. To be specific, the logistic regression scheme used here was quite carefully crafted manually, and tuned specifically for this data set. Similarly, the SVMs were fine-tuned for these models with different kernel functions. However, this was possible automatically by using the provided tuning function. Hence, it does not necessarily take a trained biometrician to achieve similar quality as with logistic regression. The separate other learning machines we employed were all rather analytically generic within their own classes.

Thus, from another perspective the conclusion might be drawn that a refined classical modelling procedure ultimately buys little appreciable performance advantage in terms of prediction error, when compared with the newer prediction engines.

In addition to being correct and easy to develop, a prognostic model needs to be clinically interpretable in order to be of practical use. One important factor in this is the transparency of the final model. Here, logistic regression models fare quite well, although maybe classification trees from CART are even more easy to understand. In contrast, prediction models from SVMs and ensemble methods are difficult to interpret.

Hence, from this data set, and more broadly from our general experience with using the prediction engines on other data, a classical prediction method such as logistic regression offers the clinician this single clear advantage: it is interpretable via the coefficients in the regression model. As a consequence, it seems that the choice for the practitioner mostly is between simple interpretability on the one hand and simple application with no assumptions and low error rates on the other hand. However, we suggest that these requirements are better understood not as competing for the same labor or intellectual space, but as goals that might be achieved in sequence. Sensibly combining the available methods as described above might yield both, low prediction error rates and transparency of the final model.

In our experience, this sequential procedure does not lead to a long cycle of testing, training or tuning, as the few most important variables selected by random forests typically do as well as the complete variable list, and by focusing on the top few variables we have acquired a useful degree of interpretability.

As the main focus of this tutorial was on the detailed explanation of the different methods, some issues could not be addressed. Firstly, we used only complete case records. However, we note that the random forest procedure has options for imputing missing components of a data vector (Liaw and Wiener, 2007). Furthermore, approaches for dealing with missing data have been developed for SVMs (Tsuda et al., 2003) which were not be studied here. Secondly, we have to emphasise that our example data set was more or less balanced with regard to outcome groups. Using unbalanced data – those having sharply unequal group sizes –, most algorithms by default aim at minimising the overall error fraction which will inevitably mean a minimisation of error fraction in the larger outcome group at the expense of the error fraction for the smaller group. If the aim is to yield similar sensitivity and specificity, either additional weight parameters or down-sampling have to be employed. Thirdly, we have only considered binary outcome data. For most machines, generalisations to multiclass and linear regression problems are straightforward. Finally, we did not investigate the issue of sample size. Thus, we are not able to answer what the required sample sizes for different methods are. However, it has been pointed out that learning machines are better able to deal with smaller sample sizes than classical regression models.

Having explained and applied both classical algorithms and learning machines to our data, we are still in search for *the optimal machine*. Also, having found a good performance of some of the learning machines, we still wonder why they do as well as they do? Further, what really lurks inside these black boxes? These questions have become a research topic in itself, and for the boosting and random forests methods this is an active area. No doubt, answers will be forthcoming, and then new methods based on the insights gained will be introduced. Meanwhile we argue that a patient-centered approach to prognosis must, first and foremost, make sound predictions, however they are generated. We do not believe any class of patients are well-served if the model is 'clinically sensible' and easily understood, but performs less well than a more complex learning machine.

Acknowledgements

We are grateful to the Members of the German Stroke Study Collaboration for data collection. It includes the following neurology departments and responsible study investigators (investigators in alphabetical order): St. Katharinen-Hospital Frechen (R. Adams), Charité Berlin (N. Amberger), Städtisches Krankenhaus München-Harlaching (K. Aulich, M.J.L. Wimmer), Klinikum Minden (J. Glahn), University of Magdeburg (M. Goertler), Krankenanstalten Gilead Bielefeld (C. Hagemeister), Klinikum München-Großhadern (G.F. Hamann, A. Müllner), Rheinische Kliniken Bonn (C. Kley), University of Rostock (A. Kloth), Benjamin Franklin University of Berlin (C. Koennecke), University of Saarland (P. Kostopoulos), Bürgerhospital Stuttgart (T. Mieck), University of Essen (G. Mörger-Kiefer, C. Weimar), University of Ulm (M. Riepe), University of Leipzig (D.S. Schneider), University of Jena (V. Willig).

The authors thank K. Kraywinkel, MD, MSc, and P. Dommes, PhD, for central data collection and management.

The authors would like to acknowledge the many helpful suggestions of the anonymous reviewers which lead to a substantial improvement of this paper.

This work was supported by the Deutsche Forschungsgemeinschaft.

References

Agresti, A. (1990) *Categorical Data Analysis*, Wiley, New York.

Altman, D.G. and Lyman, G.H. (1998) 'Methodological challenges in the evaluation of prognostic factors in breast cancer', *Breast Cancer Research and Treatment*, Vol. 52, pp.289–303.

Altman, D.G. and Royston, P. (2000) 'What do we mean by validating a prognostic model?', *Statistics in Medicine*, Vol. 19, pp.453–473.

Bagley, S.C., White, H. and Golomb, B.A. (2001) 'Logistic regression in the medical literature: standards for use and reporting, with particular attention to one medical domain', *Journal of Clinical Epidemiology*, Vol. 54, pp.979–985.

Breiman, L. (1996) 'Bagging predictors', *Machine Learning*, Vol. 24, pp.123–140.

Breiman, L. (2001) 'Random forests', *Machine Learning*, Vol. 45, pp.5–32.

Breiman, L. (2004) *Consistency for a Simple Model of Random Forests*, Technical Report No. 670, Statistical Department, University of California at Berkeley.

Breiman, L., Freidman, J., Stone, C. and Olshen, R. (1984) *Classification and Regression Trees*, Wadsworth, Belmont, CA.

Brown, L. and Li, X. (2005) 'Confidence intervals for two sample binomial distribution', *Journal of Statistical Planning and Inference*, Vol. 130, pp.359–375.

Bühlmann, P. (2003) 'Bagging, subagging and bragging for improving some prediction algorithms', in Akritas, M.G. and Politis, D.N. (Eds.): *Recent Advances and Trends in Nonparametric Statistics*, Elsevier, St Louis, pp.19–34.

Committee for Proprietary Medicinal Products (CPMP) (2000) 'Points to consider on clinical investigation of medicinal products for the treatment of acute stroke', *The European Agency for the Evaluation of Medicinal Products*, CPMP/EWP/560/98.

Concato, J., Feinstein, A.R. and Holford, T.R. (1993) 'The risk of determining risk with multivariable models', *Annals of Internal Medicine*, Vol. 118, pp.201–210.

Costanza, M.C., Paccaud, F. (2004) 'Binary classification of dyslipidemia from the waist-to-hip ratio and body mass index: a comparison of linear, logistic, and CART models', *BMC Medical Research Methodology*, Vol. 4, p.7.

Cristianini, N. and Shawe-Taylor, J. (2000) *An Introduction to Support Vector Machines and Other Kernel-Based Learning Methods*, Cambridge University Press, Cambridge, MA.

Dettling, M. (2004) 'BagBoosting for tumor classification with gene expression data', *Bioinformatics*, Vol. 20, pp.3583–3593.

Dettling, M. and Bühlmann P. (2003) 'Boosting for tumor classification with gene expression data', *Bioinformatics*, Vol. 19, pp.1061–1069.

Diaz-Uriarte, R. and Alvarez de Andres, S. (2006) 'Gene selection and classification of microarray data using random forest', *BMC Bioinformatics*, Vol. 7, p.3.

Dietterich, T.G. (2000) 'Ensemble methods in machine learning', in Kittler, J. and Roli, F. (Eds.): *First International Workshop on Multiple Classifier Systems, Lecture Notes in Computer Science*, Springer, New York, pp.1–15.

Domingos, P. and Pazzani, M. (1997) 'Beyond independence: conditions for the optimality of the simple Bayesian classifier', *Machine Learning*, Vol. 29, pp.103–130.

Drew, P.J., Ilstrup, D.M., Kerin, M.J. and Monson, J.R. (1999) 'Prognostic factors: guidelines for investigation design and state of the art analytical methods', *Surgical Oncology*, Vol. 7, pp.71–76.

Freund, Y. (2001) 'An adaptive version of the boost by majority algorithm', *Machine Learning*, Vol. 43, pp.293–318.

Freund, Y. and Schapire, R.E. (1997) 'A decision-theoretic generalization of on-line learning and an application to boosting', *Journal of Computer and System Sciences*, Vol. 55, pp.119–139.

Freund, Y. and Schapire, R.E. (1999) 'A short introduction to boosting', *Journal of Japanese Society for Artificial Intelligence*, Vol. 14, pp.771–780.

Friedman, J., Hastie, T. and Tibshirani, R. (2000) 'Additive logistic regression: a statistical view of boosting', *Annals of Statistics*, Vol. 38, pp.337–374.

German Stroke Study Collaboration (2004) 'Predicting outcome after acute ischemic stroke: an external validation of prognostic models', *Neurology*, Vol. 62, pp.581–585.

Guyon, I. and Elisseeff, A. (2003) 'An introduction to variable and feature selection', *Journal of Machine Learning Research*, Vol. 3, pp.1157–1182.

Hand, D.J. (2006) 'Classifier technology and the illusion of progress', *Statistical Science*, Vol. 21, pp.1–14.

Harrell, F.E., Lee, K.L. and Mark, D.B. (1996) 'Multivariable prognostic models: issues in developing models, evaluating assumptions and adequacy, and measuring and reducing errors', *Statistics in Medicine*, Vol. 15, pp.361–387.

Hart, A. and Wyatt, J. (1990) 'Evaluating black-boxes as medical decision aids: issues arising from a study of neural networks', *Medical Informatics (London)*, Vol. 15, pp.229–236.

Hastie, T., Tibshirani, R. and Friedman, J. (2003) *The Elements of Statistical Learning. Data Mining, Inference, and Prediction*, Springer, New York.

Hoeting, J.A. (2002) 'Methodology for Bayesian model averaging: an update', *Proceedings-Manuscripts of Invited Paper Presentations, International Biometric Conference*, Freiburg, Germany, pp.231–240.

Hoeting, J.A., Madigan, D., Raftery, A.E. and Volinsky, C.T. (1999) 'Bayesian model averaging: a tutorial', *Statistical Science*, Vol. 14, pp.382–417.

Hothorn, T. and Lausen, B. (2003) 'Double-bagging: combining classifiers by bootstrap aggregation', *Pattern Recognition*, Vol. 36, pp.1303–1309.

Hothorn, T. and Lausen, B. (2005) 'Bundling classifiers by bagging trees', *Computational Statistics and Data Analysis*, Vol. 49, pp.1068–1078.

Huang, K. and Murphy, R.F. (2004) 'Boosting accuracy of automated classification of fluorescence microscope images for location proteomics', *BMC Bioinformatics*, Vol. 5, p.78.

Koltchinskii, V. and Yu, B. (2004) 'Three papers on boosting: an introduction', *Annals of Statistics*, Vol. 32, p.12.

König, I.R., Malley, J.D., Weimar, C., Diener, H-C. and Ziegler, A. (2007) 'On behalf of the German Stroke Study Collaboration. Practical experiences on the necessity of external validation', *Statistics in Medicine*, Vol. 26, pp.5499–5511.

König, I.R., Weimar, C., Diener, H-C. and Ziegler, A. (2003) 'Vorhersage des Funktionsstatus 100 Tage nach einem ischÄamischen Schlaganfall: Design einer prospektiven Studie zur externen Validierung eines prognostischen Modells', *Zeitschrift für ärztliche Fortbildung und Qualitätssicherung*, Vol. 97, pp.717–722.

Kuhn, H.W. and Tucker, A.W. (1951) 'Nonlinear programming', in Neyman, J. (Ed.): *Proceedings of the 2nd Berkeley Symposium on Mathematical Statistics and Probabilistics*, University of California Press, Berkeley, pp.75–113.

Liaw, A. and Wiener, M. (2007) Fortran original by Breiman, L. and Cutler, A. The random Forest package, Version 4.5.18, http://cran.r-project.org/web/packages/randomForest/index.html

Lin, N., Wu, B., Jansen, R., Gerstein, M. and Zhao, H. (2004) 'Information assessment on predicting protein-protein interactions', *BMC Bioinformatics*, Vol. 5, p.154.

Mahoney, F.I., Barthel, D.W. (1965) 'Functional evaluation: the Barthel index', *Maryland State Medical Journal*, Vol. 14, pp.61–65.

McKelvey, R.D. and Zavoina, W. (1975) 'A statistical model for the analysis of ordinal level dependent variables', *Journal of Mathematical Sociology*, Vol. 4, pp.103–120.

McLaughlin, W.A. and Berman, H.M. (2003) 'Statistical models for discerning protein structures containing the DNA-binding helix-turn-helix motif', *Journal of Molecular Biology*, Vol. 330, pp.43–55.

Meir, R. and Rätsch, G. (2003) 'An introduction to boosting and leveraging', in Mendelson, S. and Smola, A. (Eds.): *Advanced Lectures on Machine Learning*, Springer, New York.

Mojirsheibani, M. (1999) 'Combining classifiers via discretization', *Journal of the American Statistical Association*, Vol. 94, pp.600–609.

Newcombe, R.G. (1998) 'Improved confidence intervals for the difference between binomial proportions based on paired data', *Statistics in Medicine*, Vol. 17, pp.2635–2650.

Newcombe, R.G. (1999) 'Interval estimation for the difference between independent proportions: comparison of eleven methods', *Statistics in Medicine*, Vol. 17, 1998, pp.873–890. *Erratum in Statistics in Medicine* 1999, Vol. 18, p.1293.

Noble, W.S. (2006) 'What is a support vector machine?', *Nature Biotechnology*, Vol. 24, pp.1565–1567.

Opitz, D. and Maclin, R. (1999) 'Popular ensemble methods: an empirical study', *Journal of Artificial Intelligence Research*, Vol. 11, pp.169–198.

Ottenbacher, K.J., Ottenbacher, H.R., Tooth, L. and Ostir, G.V. (2004) 'A review of two journals found that articles using multivariable logistic regression frequently did not report commonly recommended assumptions', *Journal of Clinical Epidemiology*, Vol. 57, pp.1147–1152.

Qu, Y., Adam, B.L., Yasui, Y., Ward, M.D., Cazares, L.H., Schellhammer, P.F., Feng, Z., Semmes, O.J. and Wright Jr., G.L. (2002) 'Boosted decision tree analysis of surface-enhanced laser desorption/ionization mass spectral serum profiles discriminates prostate cancer from noncancer patients', *Clinical Chemistry*, Vol. 48, pp.1835–1843.

R Development Core Team (2005) *R: A Language and Environment for Statistical Computing*, *R* Foundation for Statistical Computing, Vienna, URL: http://www.R-project.org

Reed, M. and Simon, B. (1980) *Methods of Modern Mathematical Physics. Vol. 1: Functional Analysis*, Academic Press, San Diego.

Roberts, L. and Counsell, C. (1998) 'Assessment of clinical outcomes in acute stroke trials', *Stroke*, Vol. 29, pp.986–991.

Royston, P. and Sauerbrei, W. (2004) 'A new approach to modelling interactions between treatment and continuous covariates in clinical trials by using fractional polynomials', *Statistics in Medicine*, Vol. 23, pp.2509–2525.

Royston, P., Sauerbrei, W. and Altman, D.G. (2000) 'Modeling the effects of continuous risk factors', *Journal of Clinical Epidemiology*, Vol. 53, pp.219–221.

Sackett, D.L., Straus, S.E., Richardson, W.S., Rosenberg, W. and Haynes, R.B. (2000) *Evidence-based Medicine*, Churchill Livingstone, New York.

Schapire, R.E. (1990) 'The strength of weak learnability', *Machine Learning*, Vol. 5, pp.197–227.

Schölkopf, B. and Smola, A.J. (2002) *Learning with Kernels*, MIT Press, Cambridge, MA.

Schuntermann, M.F. (1996) 'The international classification of impairments, disabilities and handicaps (ICIDH) – results and problems', *International Journal of Rehabilitation Research*, Vol. 19, pp.1–11.

Schwarz, D.F., Szymczak, S., Ziegler, A. and König, I.R. (2007) 'Picking single nucleotide polymorphisms in forests', *BMC Proceedings*, Vol. 1, p.S59.

Schwarzer, G., Nagata, T., Mattern, D., Schmelzeisen, R. and Schumacher, M. (2003) 'Comparison of fuzzy inference, logistic regression, and classification trees (CART). Prediction of cervical lymph node metastasis in carcinoma of the tongue', *Methods of Information in Medicine*, Vol. 42, pp.572–577.

Schwender, H., Zucknick, M., Ickstadt, K. and Bolt, H.M. (2004) 'A pilot study on the application of statistical classification procedures to molecular epidemiological data', *Toxicology Letters*, Vol. 151, pp.291–299.

Simon, R. and Altman, D.G. (1994) 'Statistical aspects of prognostic factor studies in oncology', *British Journal of Cancer*, Vol. 69, pp.979–985.

Simon, R.M., Korn, E.L., McShane, L.M., Radmacher, M.D., Wright, G.W. and Zhao, Y. (2003) *Design and Analysis of DNA Microarray Investigations*, Springer, New York.

Smolle, J. and Gerger, A. (2003) 'Tissue counter analysis of tissue components in skin biopsies: evaluation using CART (Classification and Regression Trees)', *American Journal of Dermatopathology*, Vol. 25, pp.215–222.

Strobl, C., Boulesteix, A.L., Zeileis, A. and Hothorn, T. (2007) 'Bias in random forest variable importance measures: illustrations, sources and a solution', *BMC Bioinformatics*, Vol. 8, p.25.

Tango, T. (1998) 'Re: improved confidence intervals for the difference between binomial proportions based on paired data by Robert G. Newcombe', *Statistics in Medicine*, Vol. 18, pp.3511–3513.

Tango, T. (1999) 'Re: improved confidence intervals for the difference between binomial proportions based on paired data by Robert G. Newcombe', *Statistics in Medicine*, Vol. 17, pp.2635–2650.

Tsuda, K., Akaho, S. and Asai, K. (2003) 'The EM algorithm for kernel matrix completion with auxiliary data', *Journal of Machine Learning Research*, Vol. 4, pp.67–81.

Tukey, J.D. (1977) *Exploratory Data Analysis*, Addison-Wesley, Reading, MA.

Verweij, P.J. and Van Houwelingen, H.C. (1993) 'Cross-validation in survival analysis', *Statistics in Medicine*, Vol. 12, pp.2305–2314.

Walker, A.E., Robins, M. and Weinfeld, F.D. (1981) 'The national survey of stroke. Clinical findings', *Stroke*, Vol. 12, pp.113–144.

Weimar, C., König, I.R., Kraywinkel, K., Ziegler, A. and Diener, H-C. (2004) 'On behalf of the German Stroke Study Collaboration. Age and National Institutes of Health Stroke Scale score within 6 hours after onset are accurate predictors of outcome after cerebral ischemia. Development and external validation of prognostic models', *Stroke*, Vol. 35, pp.158–162.

Weimar, C., Ziegler, A., König, I.R. and Diener H-C. (2002) 'On behalf of the German Stroke. Study collaborators. Predicting functional outcome and survival after acute ischemic stroke', *Journal of Neurology*, Vol. 249, pp.888–895.

Witten, I.H. and Frank, E. (2005) *Data Mining: Practical Machine Learning Tools and Techniques*, 2nd ed., Morgan Kaufmann: San Francisco, CA.

Wyatt, J.C. and Altman, D.G. (1995) 'Commentary: prognostic models: clinically useful or quickly forgotten?', *British Medical Journal*, Vol. 311, pp.1539–1541.

Zhang, H. and Singer, B. (1999) *Recursive Partitioning in the Health Sciences*, Springer, New York.

Zhou, X.H. and Qin, G.S. (2005) 'A new confidence interval for the difference between two binomial proportions of paired data', *Journal of Statistical Planning and Inference*, Vol. 128, pp.527–542.

Zhou, X.H. and Qin, G.S. (2007) 'A new confidence interval for the difference between two binomial proportions of paired data', *Supplement in Journal of Statistical Planning and Inference*, Vol. 137, pp.357–358.

Ziegler, A., DeStefano, A. and König, I.R., on behalf of Group 6 (2007) 'Data mining, neural nets, trees problems 2 and 3 of genetic analysis workshop 15', *Genetic Epidemiology*, Vol. 33, pp.S51–S60.

Appendix A: List of investigated variables

Table 6 includes all variables that were available in both the training and the test data set. The test data set contains centres for both temporal and external validation.

Appendix B: Some vector space details and mathematical details for support vector machines

B.1: Equation for a plane

The equation for a hyperplane $\mathcal{L}$ in $\mathbb{R}^n$ which is orthogonal, denoted: $\perp$ to some vector $\boldsymbol{\beta}$ may be derived as follows:

- Let the vectors $\boldsymbol{x}_1, \boldsymbol{x}_2$ be on $\mathcal{L}$. Then $\boldsymbol{x}_1 - \boldsymbol{x}_2$ is a line segment on $\mathcal{L}$.
- The required condition for $\boldsymbol{\beta}$ being orthogonal to $\mathcal{L}$ is then: $\boldsymbol{\beta}'(\boldsymbol{x}_1 - \boldsymbol{x}_2) = \mathbf{0}$
- Fix $\boldsymbol{x}_2$ and let $\boldsymbol{x}_1 = x$ vary. Then the equation for $\boldsymbol{x}$ on $\mathcal{L}$ is given by

$$\mathbf{0} = \boldsymbol{\beta}'(\boldsymbol{x} - \boldsymbol{x}_2) = \boldsymbol{\beta}'\boldsymbol{x} - \boldsymbol{\beta}'\boldsymbol{x}_2 =: \boldsymbol{\beta}'\boldsymbol{x} + \beta_0,$$

where $\beta_0 = -\boldsymbol{\beta}'\boldsymbol{x}_2 \in \mathbb{R}$ is scalar.

B.2: Distance from a point to a plane

Figure 12 illustrates the problem. $\boldsymbol{\beta}$ is a vector, and $\boldsymbol{x} \in \mathbb{R}^n$. We aim at finding $\boldsymbol{x}_0$ on the plane $\mathcal{L}$ so that the line segment $\boldsymbol{x} - \boldsymbol{x}_0$ parallels $\boldsymbol{\beta}$.

The angle ϑ between any two vectors $\boldsymbol{x}$ and $\boldsymbol{y}$ is given by

$$\cos(\vartheta) = \frac{\boldsymbol{x}'\boldsymbol{y}}{\|\boldsymbol{x}\| \cdot \|\boldsymbol{y}\|},$$

where $\|\boldsymbol{x}\|^2 = \boldsymbol{x}'\boldsymbol{x}$.

The line segment $\boldsymbol{x} - \boldsymbol{x}_0$ needs to be parallel to $\boldsymbol{\beta}$, yielding $\vartheta = \vartheta\big(\boldsymbol{\beta}, \boldsymbol{x} - \boldsymbol{x}_0\big) = 0$. Thus, $\cos(\vartheta) = 1$. This leads to

$$1 = \frac{\boldsymbol{\beta}'(\boldsymbol{x} - \boldsymbol{x}_0)}{\|\boldsymbol{\beta}\|\|\boldsymbol{x} - \boldsymbol{x}_0\|}$$

The distance from $\boldsymbol{x}$ to $\boldsymbol{x}_0$ which is on $\mathcal{L}$ is therefore given by

$$\|\boldsymbol{x} - \boldsymbol{x}_0\| = \frac{1}{\|\boldsymbol{\beta}\|}\big(\boldsymbol{\beta}'\boldsymbol{x} + \beta_0\big)$$

Let $f(\boldsymbol{x}) = \boldsymbol{\beta}'\boldsymbol{x} + \beta_0$. If $\|\boldsymbol{\beta}\| = 1$, then $f(\boldsymbol{x})$ is the distance from the point $\boldsymbol{x}$ to the plane $\mathcal{L}$ since $\boldsymbol{x}_0$ is on $\mathcal{L}$. If a point $\boldsymbol{x}_0$ is on $\mathcal{L}$, then the equation $\boldsymbol{\beta}'\boldsymbol{x}_0 + \beta_0 = 0$ is fulfilled. If, however, $\boldsymbol{x}_0$ is C distant from $\mathcal{L}$, then $\boldsymbol{\beta}'\boldsymbol{x}_0 + \beta_0 - C = 0$ (or $\boldsymbol{\beta}'\boldsymbol{x}_0 + \beta_0 + C = 0$).

Table 6 Variables considered in test and training data

Assessment time	*Variable*
Demographics	Age
	Gender
	Smoking status within last five years
History	Prior stroke
	Prior peripheral arterial disease
	Arterial hypertension (elevated blood pressure above 160/95 mmHg or antihypertensive medication)
	Diabetes mellitus (elevated blood glucose or elevated HbA1c or antidiabetic medication)
At admission	Overall functional impairments rated on Modified Rankin Scale
	Baseline neurological impairments rated on the National Institute of Health Stroke Scale with:
	Total score
	Level of consciousness
	Questions
	Commands
	Best gaze
	Visual
	Facial palsy
	Left arm paresis
	Right arm paresis
	Left leg paresis
	Right leg paresis
	Limb ataxia
	Sensory
	Best language
	Dysarthria
	Extinction and inattention
Within 72 hours after admission	Lowering of elevated blood glucose
	Fever (rectal temperature rise to >38)
	Neurological complications: recurrent cerebral ischemia, symptomatic parenchymal bleeding, or epileptic seizure
	Other medical complications: hypertensive crisis, cardical arrhythmia, lung edema/cardiac failure, pneumonia, venous thrombosis, pulmonary embolism, or peripheral bleeding
	Localisation of infarction in cerebral imaging with:
	Anterior cerebral artery
	Middle cerebral artery
	Lenticulostriate arteries
	Thalamic arteries
	Posterior cerebral artery
	Cerebellar arteries
	Brain stem arteries
	Borderline middle/posterior cerebral arteries
	Borderline anterior/middle cerebral arteries
	Long perforating arteries

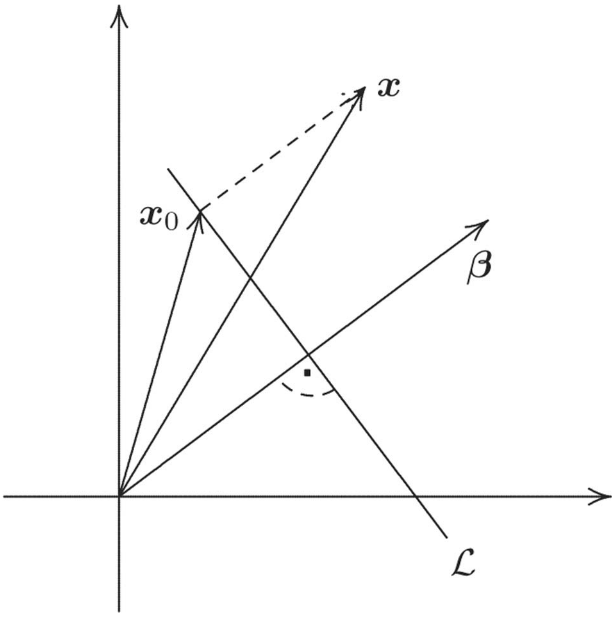
B.3: Restating the optimisation problem in the separable case

We show that the optimisation problem of equation (1) for the separable case can be restated as the solution of:

$$\begin{aligned}&\min_{\boldsymbol{\beta},\beta_0} && \|\boldsymbol{\beta}\| \\ &\text{subject to} && y_i(\boldsymbol{x}'\boldsymbol{\beta}+\beta_0)\geq C, \quad i=1,\dots,n\end{aligned}$$

In this restatement of equation (1), the norm constraint on $\boldsymbol{\beta}$ can be dropped by setting $C = 1/\|\boldsymbol{\beta}\|$. This yields a convex optimisation problem, and the solution is outlined in Appendix C.

Figure 12 The linear algebra of an affine hyperplane $\mathcal{L}$. The point $\boldsymbol{x}_0$ on the hyperplane $\mathcal{L}$ is found in such a way that the segment $\boldsymbol{x} - \boldsymbol{x}_0$ parallels $\boldsymbol{\beta}$



The proof of the restatement is as follows: We want $f(\boldsymbol{x}) = \boldsymbol{\beta}'\boldsymbol{x} + \beta_0$ such that for each pair of observations $(y_i, \boldsymbol{x}_i)$:

$$\boldsymbol{\beta}'\boldsymbol{x}_i + \beta_0 \geq C$$

with the constraint $\|\boldsymbol{\beta}\| = 1$ and a constant C to be maximised. For this C, every data point $\boldsymbol{x}_i$ is at least C distant from the hyperplane so that

$$\boldsymbol{\beta}'\boldsymbol{x}_i + \beta_0 \geq C \quad \text{or} \quad \boldsymbol{\beta}'\boldsymbol{x}_i + \beta_0 \leq -C.$$

This is exactly $y_i\big(\boldsymbol{\beta}'\boldsymbol{x}_i + \beta_0\big) \geq C$ for $y_i = \pm1$.

Suppose we let $\|\boldsymbol{\beta}\|$ vary. Then the required equation is

$$y_i\left(\frac{\boldsymbol{\beta}'}{\|\boldsymbol{\beta}\|}\boldsymbol{x}_i - \frac{\boldsymbol{\beta}'}{\|\boldsymbol{\beta}\|}\boldsymbol{x}_2\right) \geq C$$

since $\beta_0 = \boldsymbol{\beta}'\boldsymbol{x}_2$ from above.

This equation is true for all $\boldsymbol{\beta}$ so that we can set $\|\boldsymbol{\beta}\| = \frac{1}{C}$, and we get $y_i\big(\boldsymbol{\beta}'\boldsymbol{x}_i + \beta_0\big) \geq 1$, where $\max C \Leftrightarrow \min \|\boldsymbol{\beta}\|$. In the separable case, the equation for a SVM is just $\min \|\boldsymbol{\beta}\|$ for some $\boldsymbol{\beta}, \beta_0$ such that $y_i(\boldsymbol{\beta}'\boldsymbol{x}_i + \beta_0) \geq 1$.

B.4: Solving the maximisation problem in the non-separable case

As in the separable case, the norm constraint on $\boldsymbol{\beta}$ can be eliminated by letting $C = 1/\|\boldsymbol{\beta}\|$, so that we aim at finding β_0 and $\boldsymbol{\beta}$ such that

$$\begin{array}{ll} \min\limits_{\boldsymbol{\beta},\beta_0} & \|\boldsymbol{\beta}\| \\ \text{subject to} & y_i(\boldsymbol{x}'\boldsymbol{\beta} + \beta_0) \geq 1 - \xi_i, \\ \text{and to} & \xi_i \geq 0, \quad \sum\limits_{i=1}^{n} \xi_i \leq \gamma, \quad i = 1, \ldots, n. \end{array} \tag{8}$$

The solution for the optimisation problem (8) can be obtained by a quadratic programming algorithm using Lagrange multipliers. The Lagrange primal function is given by

$$L_P = \frac{1}{2}\|\boldsymbol{\beta}\|^2 + \gamma\sum_{i=1}^{n}\xi_i - \sum_{i=1}^{n}\alpha_i(y_i(\boldsymbol{x}'\boldsymbol{\beta} + \beta_0) - (1 - \xi_i)) - \sum_{i=1}^{n}\delta_i\xi_i \tag{9}$$

In equation (9), the first condition replaces the constant, the second the condition of the hyperplane, and the third the positivity of the slack variables. If equation (9) is minimised w.r.t. $\beta_0, \boldsymbol{\beta}$ and ξ_i, one obtains

$$\begin{aligned} \boldsymbol{\beta} &= \sum_{i=1}^{n}\alpha_i y_i \boldsymbol{x}_i, \\ 0 &= \sum_{i=1}^{n} y_i, \\ \alpha_i &= \gamma - \delta_i, \quad \text{for all } i \\ \alpha_i &\geq 0, \quad \delta_i \geq 0, \quad \xi_i \geq 0, \quad \text{for all } i \end{aligned}$$

after setting the derivatives to 0. The important simplification can now be carried out. It is the step from the Lagrange primal function to its dual which gives a lower bound on equation (9):

$$L_D = \sum_{i=1}^{n}\alpha_i - \frac{1}{2}\sum_{i=1}^{n}\sum_{j=1}^{n}\alpha_i\alpha_j y_i y_j \boldsymbol{x}_i'\boldsymbol{x}_j. \tag{10}$$

Equation (10) is maximised subject to $0 \leq \alpha_i \leq \gamma$ and $\sum_{i=1}^{n} \alpha_i y_i = 0$. All restraints together with the Karush-Kuhn-Tucker conditions of optimisation (Kuhn and Tucker, 1951) lead to the solution for $\boldsymbol{\beta}$:

$$\hat{\boldsymbol{\beta}} = \sum_{i=1}^{n} \hat{\alpha}_i y_i \boldsymbol{x}_i.$$

$\hat{\alpha}_i$ are only nonzero for those observations i for which the constraints $y_i \left(\boldsymbol{x}_i' \boldsymbol{\beta} + \beta_0 \right) \geq 1 - \xi_i$ are exactly met. The observations with positive $\hat{\alpha}_i$ are called the *support vectors*.